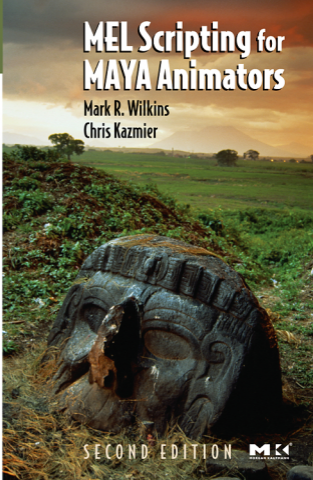




MEL Scripting for MAYA Animators

Mark R. Wilkins
Chris Kazmier



SECOND EDITION



MEL Scripting for Maya Animators, 2e

Reading and following the lessons of this book provides one of the best ways for a casual Maya user to elevate their skills to a professional level. The fundamental techniques developed in MEL Scripting for Maya Animators are critical for visual effects artists to learn.

—Scott Stokdyk, Visual Effects Supervisor, Sony Pictures Imageworks

While the first edition opened the doors to MEL scripting for interested Maya users, this expanded and updated second edition delves deeper into important programming concepts, and new features in Maya released since the original edition. No longer just for the first steps in MEL programming, this edition takes the reader into the depths of advanced topics such as user interface layout and web panels that are sure to make this book a frequently used and a welcome addition to any technical director's bookshelf.

—Doug Cooper, Visual Effects Supervisor, DreamWorks Animation

MEL Scripting for Maya Animators is the set of keys you need to get under the hood of Maya. The book is well written for both technical and non-technical animators. It is an essential tool in making sophisticated animation not only possible but also practical.

—Henry LaBounta, Senior Art Director, Electronic Arts

MEL Scripting for Maya Animators

Second Edition

The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling

- Complete Maya Programming Volume II:
An In-depth Guide to 3D Fundamentals, Geometry, and Modeling*
David A. D. Gould
- High Dynamic Range Imaging:
Data Acquisition, Manipulation, and Display*
Erik Reinhard, Greg Ward, Sumanta Pattanaik, and Paul Debevec
- MEL Scripting for Maya Animators, Second Edition*
Mark R. Wilkins and Chris Kazmier
- Advanced Graphics Programming Using OpenGL*
Tom McReynolds and David Blythe
- Digital Geometry:
Geometric Methods for Digital Picture Analysis*
Reinhard Klette and Azriel Rosenfeld
- Digital Video and HDTV:
Algorithms and Interfaces*
Charles Poynton
- Real-Time Shader Programming*
Ron Fosner
- Complete Maya Programming:
An Extensive Guide to MEL and the C++ API*
David A. D. Gould
- Texturing & Modeling:
A Procedural Approach, Third Edition*
David S. Ebert, F. Kenton Musgrave, Darwyn Peachey, Ken Perlin, and Steven Worley
- Geometric Tools for Computer Graphics*
Philip Schneider and David H. Eberly
- Understanding Virtual Reality:
Interface, Application, and Design*
William B. Sherman and Alan R. Craig
- Jim Blinn's Corner:
Notation, Notation, Notation*
Jim Blinn
- Level of Detail for 3D Graphics*
David Luebke, Martin Reddy, Jonathan D. Cohen, Amitabh Varshney, Benjamin Watson, and Robert Huebner
- Pyramid Algorithms:
A Dynamic Programming Approach to Curves and Surfaces for Geometric Modeling*
Ron Goldman
- Non-Photorealistic Computer Graphics:
Modeling, Rendering, and Animation*
Thomas Strothotte and Stefan Schlechtweg
- Curves and Surfaces for CAGD:
A Practical Guide, Fifth Edition*
Gerald Farin
- Subdivision Methods for Geometric Design:
A Constructive Approach*
Joe Warren and Henrik Weimer
- Computer Animation: Algorithms and Techniques*
Rick Parent
- The Computer Animator's Technical Handbook*
Lynn Pocock and Judson Rosebush
- Advanced RenderMan:
Creating CGI for Motion Pictures*
Anthony A. Apodaca and Larry Gritz
- Curves and Surfaces in Geometric Modeling:
Theory and Algorithms*
Jean Gallier
- Andrew Glassner's Notebook:
Recreational Computer Graphics*
Andrew S. Glassner
- Warping and Morphing of Graphical Objects*
Jonas Gomes, Lucia Darsa, Bruno Costa, and Luiz Velho
- Jim Blinn's Corner:
Dirty Pixels*
Jim Blinn
- Rendering with Radiance:
The Art and Science of Lighting Visualization*
Greg Ward Larson and Rob Shakespeare
- Introduction to Implicit Surfaces*
Edited by Jules Bloomenthal
- Jim Blinn's Corner:
A Trip Down the Graphics Pipeline*
Jim Blinn
- Interactive Curves and Surfaces:
A Multimedia Tutorial on CAGD*
Alyn Rockwood and Peter Chambers
- Wavelets for Computer Graphics:
Theory and Applications*
Eric J. Stollnitz, Tony D. DeRose, and David H. Salesin
- Principles of Digital Image Synthesis*
Andrew S. Glassner
- Radiosity & Global Illumination*
François X. Sillion and Claude Puech
- User Interface Management Systems:
Models and Algorithms*
Dan R. Olsen, Jr.
- Making Them Move: Mechanics, Control, and Animation of Articulated Figures*
Edited by Norman I. Badler, Brian A. Barsky, and David Zeltzer
- Geometric and Solid Modeling: An Introduction*
Christoph M. Hoffmann
- An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*
Richard H. Bartels, John C. Beatty, and Brian A. Barsky

MEL Scripting for Maya Animators

Second Edition

**Mark R. Wilkins
Chris Kazmier**

with a contribution by
Stephan Osterburg



AMSTERDAM • BOSTON • HEIDELBERG • LONDON
NEW YORK • OXFORD • PARIS • SAN DIEGO
SAN FRANCISCO • SINGAPORE • SYDNEY • TOKYO
Morgan Kaufmann Publishers is an imprint of Elsevier



Senior Editor
Publishing Services Manager
Project Manager
Assistant Editor
Editorial Assistant
Cover Designer
Cover Image

Tim Cox
Simon Crump
Brandy Lilly
Richard Camp
Jessica Evans
Hannus Design
Guatemala, Pacific Slope, Finca El Baul, half-buried
stone head. Copyright Getty Images/Stone
Collection/David Hiser.
SPI Publisher Services
Dartmouth Publishing Services
SPI USA
SPI USA
SPI USA
The Maple-Vail Book Manufacturing Group
Phoenix Color

Composition
Technical Illustration
Copyeditor
Proofreader
Indexer
Interior printer
Cover printer

Morgan Kaufmann Publishers is an imprint of Elsevier.
500 Sansome Street, Suite 400, San Francisco, CA 94111

This book is printed on acid-free paper.

© 2005 by Elsevier Inc. All rights reserved.

Designations used by companies to distinguish their products are often claimed as trademarks or registered trademarks. In all instances in which Morgan Kaufmann Publishers is aware of a claim, the product names appear in initial capital or all capital letters. Readers, however, should contact the appropriate companies for more complete information regarding trademarks and registration.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means—electronic, mechanical, photocopying, scanning, or otherwise—without prior written permission of the publisher.

Permissions may be sought directly from Elsevier's Science & Technology Rights Department in Oxford, UK: phone: (+44) 1865 843830, fax: (+44) 1865 853333, e-mail: permissions@elsevier.co.uk. You may also complete your request on-line via the Elsevier homepage (<http://elsevier.com>) by selecting "Customer Support" and then "Obtaining Permissions."

Library of Congress Cataloging-in-Publication Data

Wilkins, Mark R.

MEL scripting for Maya animators / Mark R. Wilkins and Chris Kazmier.—2nd ed.

p. cm.

Includes index.

ISBN 0-12-088793-2

1. Computer animation. 2. Maya (Computer file) 3. Computer graphics. I. Kazmier, Chris. II. Title.

TR897.7.W555 2005

006.6'96—dc22

2005015610

ISBN: 0-12-088793-2

For information on all Morgan Kaufmann publications,
visit our Web site at www.mkp.com or www.books.elsevier.com

Printed in the United States of America

05 06 07 08 09 5 4 3 2 1

Working together to grow
libraries in developing countries

www.elsevier.com | www.bookaid.org | www.sabre.org

ELSEVIER

BOOK AID
International

Sabre Foundation

For my grandfather, Martell Otos, and my grandmother, Roberta Kelly Otos.

M.R.W.

For my parents, Leonard and Lorraine Kazmier.

C.K.

About the Authors

Mark R. Wilkins is a technical director at DreamWorks Animation SKG, where he helped develop a production pipeline using Maya for effects and character animation. Mark also provides training and technical assistance to animators using Maya. He previously worked at Walt Disney Feature Animation in a variety of positions including software engineer and scene setup supervisor. He has contributed to a number of films, including *Dinosaur*, *Mission: Impossible 2*, *Minority Report*, and *Madagascar*. Mark holds a degree in physics from Harvey Mudd College.

Chris Kazmier is a senior technical director at Sony Pictures Imageworks, where he creates computer-generated effects for live-action films. He has worked on projects ranging from *The Haunted Mansion* to Sony's first all 3D feature animation *Open Season*. Previously, Chris worked at DreamWorks on *Sinbad* and at PDI/DreamWorks on the Intel Aliens ad campaign. Credits also include Fox Animation Studio's *Titan AE* and *Anastasia*.

Contents

Preface xvii
Special Acknowledgment xviii

Chapter 1 Maya Under the Hood 1

In this chapter you will learn 1
Why Look Under the Hood? 1
The Dependency Graph, Attributes, and Connections 2

Example 1: Using the Hypergraph to Explore the Dependency Graph 8

Transform Hierarchy and Parent/Child Relationships 13
Examining the Hierarchy 15
Transform and Shape Nodes 15

Example 2: Exploring Transform and Shape Nodes, Instancing, and History 17

MEL and Maya's User Interface 20
What to Remember About How Maya Works Behind the Scenes 20

Chapter 2 The Basics of MEL Commands 23

In this chapter you will learn 23
Can I Use MEL Without Scripting? 23
Command Line and Command Feedback Line 24
Command Shell 25
Script Editor 25

Script Editor Versus Command Shell	27
Script Editor's Messages as MEL Code	27
Making a Shelf Button for a MEL Script	29
Saving a MEL Script	29
Seductive Dangers of the Status Message Area	30
The whatIs Command	31
Basic Structure of MEL Commands	32
Where to Find Information About Maya and MEL on the Internet	33
Newsgroups	34
How to Use MEL Scripts Found on the Internet	34
What to Remember About How to Use MEL Without Writing Scripts	35

Chapter 3 Using Expressions 37

In this chapter you will learn	37
What Is an Expression?	37
How Does an Expression Work?	38
Equals Sign: Equality and Assignment	38
How Maya Implements Expressions	40
Is Maya's Expression Language the Same as MEL?	41
When (and When Not) to Use an Expression	42
Defining Relationships Between Attributes	43
What Is Operator Precedence?	44
Walkthrough of Maya's Expression Language	45
Definitions of Variables	46
Computing the Values of Attributes	49
Assigning Computed Values	49

Example 1: Eyes 50

Analyzing the Problem	51
Planning the Eyes' Animation Controls	54
Writing the Expression	54
What to Remember About Using Expressions	60

Chapter 4 Controlling Particles with Expressions 61

In this chapter you will learn	61
Two Kinds of Particle Object Attributes: Per Object and Per Particle	61
All About Vectors	62
Two Kinds of Expressions: Ordinary and Particle	68

Example 1:	Ordinary Expressions and a Newton Field	68
Example 2:	A Simple Particle Expression	73
	A Few Hints for Efficient Particle Expressions	80
Example 3:	Helical Particles Around a Curve	80
	What to Remember About Particle Expressions in Maya	93
Chapter 5	Problem Solving with MEL Scripting	95
	In this chapter you will learn	95
	MEL's Role in Maya: Building Scenes	95
	Strategies for Planning MEL Applications	97
	The Simplest User Interface	98
	Creating, Editing, and Querying Nodes in MEL	100
	Adding, Setting, and Getting Values of Attributes in MEL	101
	Connecting Attributes in MEL	102
	Creating and Connecting Expression Nodes in MEL	103
Example 1:	Using MEL to Automate Setup for Spiral Particles	104
	What to Remember About Writing MEL Scripts	112
Chapter 6	Variables and Data Types	115
	In this chapter you will learn	115
	Declaring Variables (and Not Declaring Them)	115
	Environment Variables	121
	MEL Statements and Type Checking	121
	Simple and Aggregate Data Types	122
	What to Remember About Variables and Data Types in MEL	134
Chapter 7	Using MEL Commands	137
	In this chapter you will learn	137
	What Is a MEL Command?	137
	Structure of a MEL Command	138
	Using MEL Commands in MEL Scripts	141
	Avoid Using MEL Commands in Expressions	143
	What to Remember About Using MEL Commands	143

Chapter 8 Manipulating Nodes in MEL 145

- In this chapter you will learn 145
- Using ls Command to Find Nodes by Name or Other Properties 145
- Using Select Command to Manage Object Selection 148
- Creating Nodes in a Maya Scene 150
- Finding a Node's Parents and Children 151
- Finding Information on Node Connections 152
- About Maya's Node Types and the Node and Attribute Reference 153
- What to Remember About Managing Nodes in MEL 154

Chapter 9 Controlling the Flow of Execution 155

- In this chapter you will learn 155
- Controlling the Flow of Script Execution 155
- Basic Conditional Operations: if-else and switch 157
- Loops 167
- What to Remember About Controlling the Flow of Execution in MEL 170

Chapter 10 Procedures and Functions 171

- In this chapter you will learn 171
- Top-Down Design 171

Example 1: A Trip to the Grocery Store 172
What Are Procedures and Functions? 174

Example 2: Geometry-Constrained Locators 178

Example 3: Recursive Antenna 182
What to Remember About Procedures, Functions,
and Top-Down Design in MEL 186

Chapter 11 Naming Nodes, Scripts, and Variables 189

- In this chapter you will learn 189
- Why Naming Conventions Are Important 189
- Naming Scripts 190
- Naming Variables 191
- Naming Nodes 192

- Example 1: Adding a Name Prefix to Objects in a Hierarchy 195
- Example 2: Changing Name Prefixes in a Hierarchy 197
- What Are Namespaces? 200
 - Strategies for Using Namespaces 203
 - What to Remember About Naming Scripts, Variables, and Nodes 203

Chapter 12 Designing MEL User Interfaces 205

- In this chapter you will learn 205
- What Is a User Interface? 205
- What Maya Users Expect to See from a MEL Script 206
- Questions to Answer Before Designing a User Interface 208
- Designing and Testing a User Interface 209
- Structure of a Dialog Box 210
- What to Remember About Designing User Interfaces in MEL 212

Chapter 13 Simple MEL User Interfaces 213

- In this chapter you will learn 213
- Collecting Information from Users 213
- Validating User Input: When and Why 214
- Asking for Confirmation with `confirmDialog` 216
- Asking User for Text String with `promptDialog` 217
- Asking User to Pick File or Directory with `fileDialog` 218
- Handling Warnings and Errors with `Warning` and `Error` Commands 219
- Using Regular Expressions and `match` to Validate Data 219
- How Regular Expressions Work 220
- Validating Integers 222
- Validating Floating-Point Numbers 223
- Validating Object Names (Without Namespaces) 224

Example 1: Simple Dialogs and Input Validation 225

- What to Remember About Simple MEL User Interfaces and Input Validation 230

Chapter 14 Custom Dialog Boxes 231

- In this chapter you will learn 231
- How to Structure a Script That Uses a Custom Dialog Box for Input 231

Dialog Boxes and Their Contents 232

Example 1: Making the Example Dialog Box 235

Common Types of Controls 243

Common Types of Layouts 249

Example 2: Dialog Box for Making Geometric Primitives 252

What to Remember About Building Custom Dialog Boxes
in MEL 256

Chapter 15 Making Advanced Dialog Boxes with formLayout 259

In this chapter you will learn 259

Why Use formLayout? 259

Planning a Dialog Box for formLayout 260

Using formLayout: Overview 261

Using formLayout: Defining Placement Rules
for UI Objects 264

Example: Implementing a Dialog Box with formLayout 266

What to Remember About Making Dialog Boxes
with formLayout 270

Chapter 16 Making Advanced Dialog Boxes with Web Panels 271

In this chapter you will learn 271

What You Need to Know Before You Proceed 271

What Are Web Panels? 272

Learning Web Authoring 273

How a Dialog Box Built with Web Panels Works 274

Planning a Dialog Box for Web Panels 275

Creating a Web-Based Dialog Box for Maya 275

Launching a Web-Based Dialog Box from MEL 278

Example 1: Implementing a Dialog Box using JavaScript 283

Ideas for Dialog Boxes Built with Web Panels 293

What to Remember about Making Dialog Boxes with Web Panels 294

Chapter 17 Improving Performance with Utility Nodes 295

In this chapter you will learn 295
What Is a Utility Node? 295
When Should You Consider Using a Utility Node? 296
How to Create and Connect a Utility Node 297

Example 1: Using the plusMinusAverage Node to Find the Midpoint
Between Two Locators 300

Common Utility Nodes 301
What to Remember About Improving Performance
With Utility Nodes 302

Chapter 18 Installing MEL Scripts 303

In this chapter you will learn 303
Installing a Script to Make It Available in All Scenes 303
Installing a Script to Run When Maya Starts 304
Installing a Script into a Script Node in a Scene 304
Installing Custom Menus 306
Managing Button Shelves and Creating Custom Shelf Icons 306
What to Remember About Installing MEL Scripts 307

Chapter 19 Examples Using MEL with Particle Dynamics 309

Example 1: Introduction to Particle Goals 309

Example 2: Particle Goals on a Surface 315

Example 3: Using Goals on Multiple Surfaces 329

Example 4: Using Goals on Surfaces, Part 2 338

Chapter 20 Examples Using MEL with Solid Body Dynamics 351

Example 1: Particle Collisions 351

Example 2: Collision Events 361

Example 3: Collisions Between Objects in Solid Dynamics 374

Example 4: Solid Dynamics and Particles 394

Chapter 21 Example of a Simple Crowd System 409

Example 1: Creating a Vehicle 409

Example 2: Vehicle Interaction 424

Example 3: Vehicle Environment 437

Example 4: Fine Tuning and Completing the Script 449

Full Script Reference: crowdSystem.mel 473

Chapter 22 Examples Using MEL in Character Rigging 485

Example 1: Character Controls 485

Example 2: Building a Character User Interface 499

Full Text of mrBlahControls.mel 511

Preface

In the last three years, the pace of updates to Maya has accelerated. New features seem to appear every few months. However, the foundation of Maya, consisting of MEL and the dependency graph, remains largely unchanged.

This second edition serves to bring *MEL Scripting for Maya Animators* up to date with the interface changes that have taken place between Maya 4.5, with which the first edition was written, and Maya 6. Certain regrettable omissions, such as a chapter on formLayout, have been corrected. Known errors have been fixed. Also, new chapters on utility nodes and Maya's Web Panel feature provide some new ideas for how to use MEL in one's applications.

More than anything else, feedback from people who have used the book to learn or teach MEL has spurred us on. To the contributors of these thoughts and perspectives, thank you!

Acknowledgments

Along the way to getting this book into your hands we've had tremendous help from others, including Aron Warner, Susan Rogers, and Ken Pearce, who provided essential help getting the project started. As we wrote, our reviewers, Ed Gavin, Kate LaBore, Chris Rock, and Doug Cooper, worked long, hard hours helping us refine our work.

Our editorial staff at Morgan Kaufmann, particularly Belinda Breyer, Mona Buehler, Diane Cerra, and Cheri Palmer have paid for our book's quality with eternal vigilance. It's been a privilege to work with them.

For the second edition, Tim Cox and Richard Camp have been endlessly patient and supportive.

Finally, of course, we wish to thank our family and friends for their relentless support and the time that we've taken away from them to put into this book.

Special Acknowledgment

Thanks especially to Stephan Osterburg, our friend and colleague, for contributing a complete and well-documented character setup for the examples in Chapter 19. These character setup examples contribute enormously to the value of the book for those interested in character applications.

Mark R. Wilkins
Chris Kazmier
2/2005

Maya Under the Hood

In this chapter you will learn

- That, to Maya, your scene is nothing more than a collection of nodes and connections, which together compose the *dependency graph*.
- What a *dependency graph node* is and how it relates to what you see in the 3D viewer.
- That the Channel Box and Attribute Editor windows are the Maya interface's most commonly used tools to manipulate the attributes of dependency graph nodes.
- That certain nodes, but not all, are part of the *transform hierarchy*, which establishes spatial relationships among objects.
- How to use the Outliner, the Hypergraph, and the Connection Editor to examine how nodes fit into the dependency graph and the transform hierarchy.
- How common tasks, such as creating objects, setting animation keyframes, and modeling NURBS surfaces, change the dependency graph.
- That Maya's graphic user interface, including menus, toolbars, the timeline, and so on, are built and can be modified with MEL.

Why Look Under the Hood?

One reason some animators find MEL scripting tricky to learn is that it requires an understanding of many details of how the pieces of a scene work together, details that the Maya user interface works hard to hide. This chapter opens up many of those details and relates them to what a Maya animator sees in the interface when working without scripting.

Knowing how a Maya scene fits together is important, even when you work entirely without MEL scripting. Thus, if you're an experienced Maya animator, you will already know much of what this chapter describes. However, keeping track of your Maya scene's components and how they fit together becomes critical when developing MEL scripts because, unlike the user interface, MEL forces you to work frequently with your Maya scene at this lower level, doing much less to hide its complexity from you.

The Dependency Graph, Attributes, and Connections

Internally, every part of a scene, whether it is 3D geometry, animation, an expression relationship, a light, a texture, or the arguments that were used to create an object with history, is represented as one or more *nodes*, or, more completely, *dependency graph* or *DG* nodes. Each node has a set of *attributes*, each of which stores a characteristic of the thing the node represents. All of these nodes together with all of their connections are called the dependency graph or the *scene graph* (Figure 1.1).

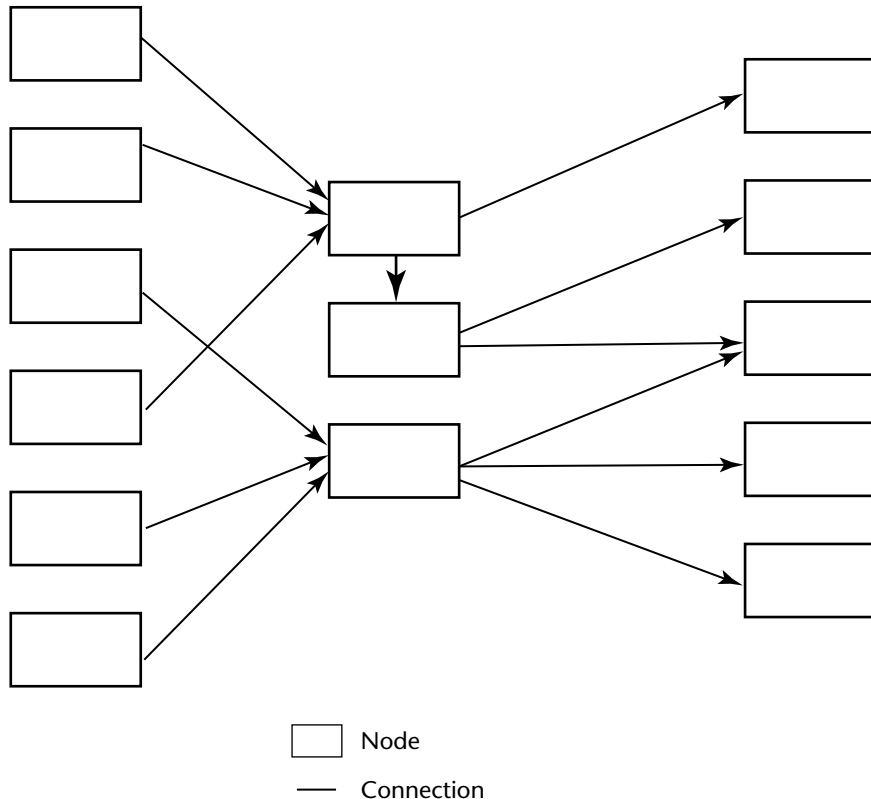


Figure 1.1 The dependency graph.

One useful source of information about DG nodes is Maya's *Node and Attribute Reference*, which is available from the Help menu. This document describes most, if not all, of the nodes that are built into Maya, along with their attributes and what function they serve in a scene.

Attributes themselves have characteristics that control how you can manipulate them. Attributes may be *locked*, which prevents them from being changed. They can be marked *keyable*, which permits you to animate them by setting keyframes, or *nonkeyable*, in which case you can't.

Also, each attribute has a *data type*, which specifies what kind of information the attribute can store. In Maya, attributes can store

- *Integer* numbers (with no fractional part).
- *Floating-point* numbers (with a fractional part).
- *Strings*, which can be a combination of text and numbers.
- *Boolean* values, which are on/off or true/false values.
- *Enumerated* values, which store a value selected from a list of choices defined when the attribute is created.
- Also, some attributes store collections of data of the above types, including *arrays*, *vectors*, and *matrices*.

One important use of MEL is to create and connect DG nodes that create a particular result when your scene is played back. At first, a good way to learn how nodes and connections work to create animation is to animate by hand, then to examine the nodes and connections that Maya has created while you have done so. Later, as we get further into the MEL language, you will learn how to build networks of nodes and attribute connections with scripts. Seeing what Maya does when you animate without scripting can serve as an important source for ideas about how to script complex tasks.

The *Channel Box*, part of the standard Maya layout, displays the one or more nodes (in this instance `directionalLight1` and `directionalLightShape1`) that make up the selected object (Figure 1.2).

The Channel Box displays only those attributes that are keyable, because those are the attributes that are most frequently edited while you work. Editing other attributes is usually done through the *Attribute Editor* (Figure 1.3). Even the Attribute Editor, though, does not display every attribute. Certain attributes can be manipulated only through MEL scripts and expressions.

The Attribute Editor displays the selected node's attributes as a series of groups that can be expanded by clicking on the arrow button to their left. In the example shown in Figure 1.3, Directional Light Attributes is one such group. Also, connected nodes appear as tabs at the top of the Attribute Editor to allow you easy access to other nodes related to the one you are editing.

The Attribute Editor allows you to add your own attributes as well, using the choices on the Attributes menu. How custom attributes can be useful in expressions and MEL is discussed in later chapters.

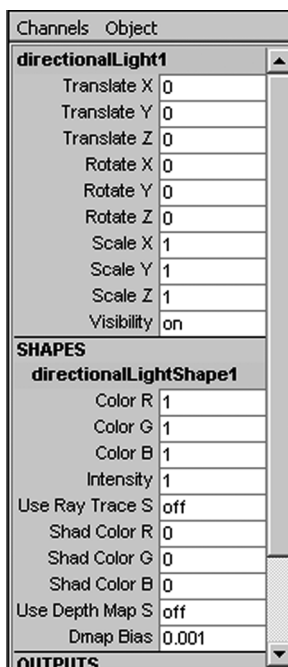


Figure 1.2 The Channel Box.

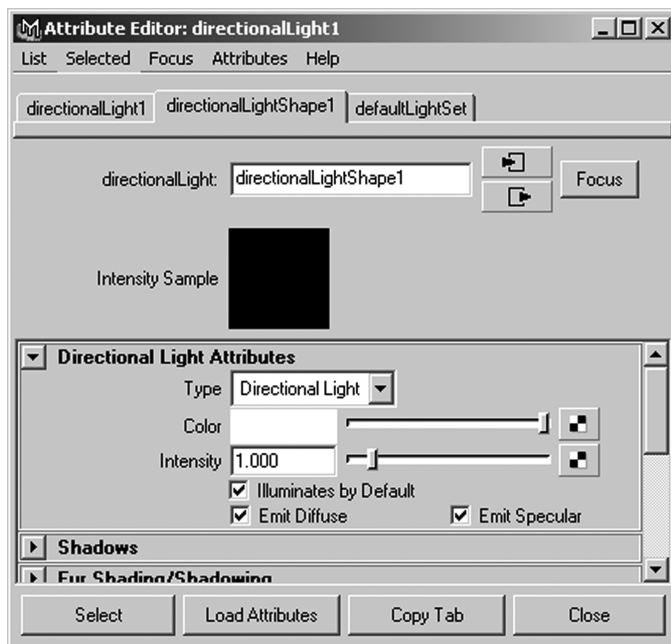


Figure 1.3 Attribute Editor.

A final important characteristic of attributes is that they can be *connected* to each other. Connecting two attributes forces the value of one attribute to remain the same as the value of another attribute. These connections are directional, meaning, for example, that if you connect one attribute to another you can change the first attribute all you like and the second will follow, but the second attribute cannot be changed because its value is being driven by the connection with the first. You can set up connections between nodes with the *Connection Editor* (Figure 1.4).

This characteristic of connection is what gives the dependency graph its name. It's a "graph" because that's a term for a network of connected nodes, and "dependency" refers to the way that each connected node depends on the values of the nodes that connect to it. Nodes whose attributes' values connect to the current node are *upstream* nodes, and nodes that depend on the current node are *downstream* nodes. The idea of a scene graph like Maya's dependency graph is common in computer animation systems; 3D Studio Max and Softimage, for example, each use scene graph structures.

The most useful tool for seeing interconnections between multiple nodes is the Hypergraph window (Figure 1.5), which you can reach by choosing Window > Hypergraph . . . from the main menu. With the Hypergraph you

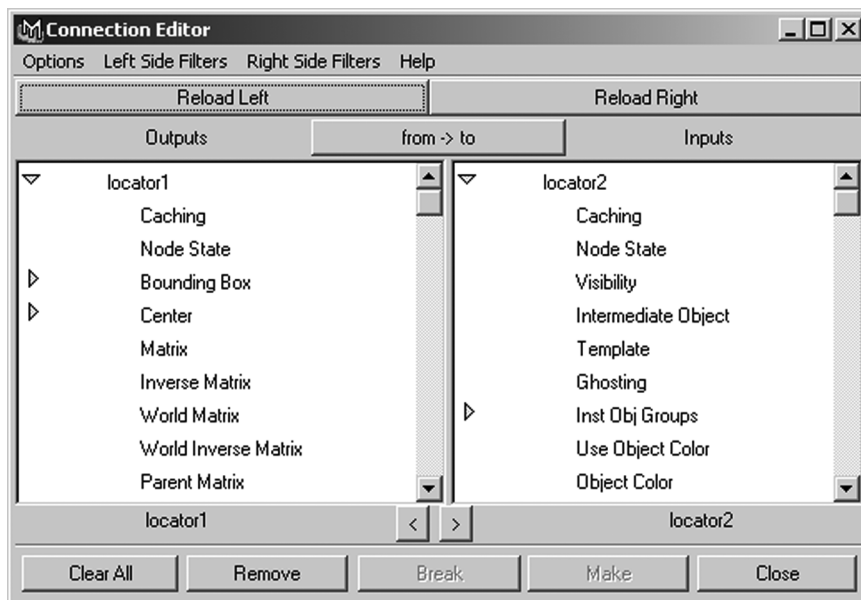


Figure 1.4 Connection Editor.

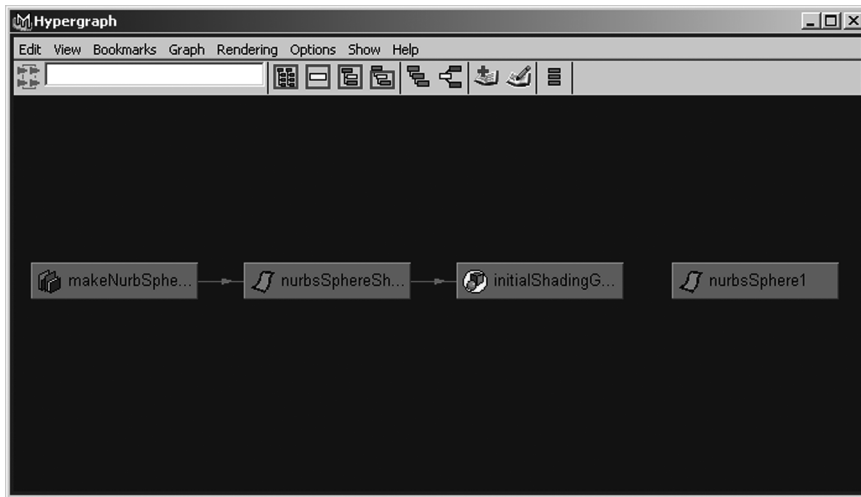


Figure 1.5 Viewing connections with the Hypergraph.

can see the connections that have been made to the selected node by selecting the node and choosing Graph > Up and Downstream Connections from the window's menus.

Note that it is not possible to view the entire scene in Up and Downstream Connections mode. Because there can be so many nodes and connections in even a relatively simple scene, this view in the Hypergraph displays only the connections that lead to the selected object so that working in the Hypergraph remains manageable. While viewing nodes in this mode, you can move the pointer over the arrows that represent connections to see what attributes are connected.

To see how a simple Maya scene is represented as a dependency graph, let's examine what happens when you animate a bouncing ball. For the sake of simplicity, let's assume the ball is only moving along one axis.

As a first pass at animating this ball (Figure 1.6), you might set three keyframes on the ball's vertical motion at, say, 0 frames, 10 frames, and 20 frames into the animation.

As you work, each action you perform creates nodes and connections behind-the-scenes. First, as seen in Figure 1.7, creating the sphere makes three connected nodes.

Then, setting the animation keyframes creates and connects another node (Figure 1.8), this time an animation curve that drives the `translateY` attribute of `nurbsSphere1`. When you play back the scene, Maya's `time1` node, the scene's clock, tells the animation curve which frame num-

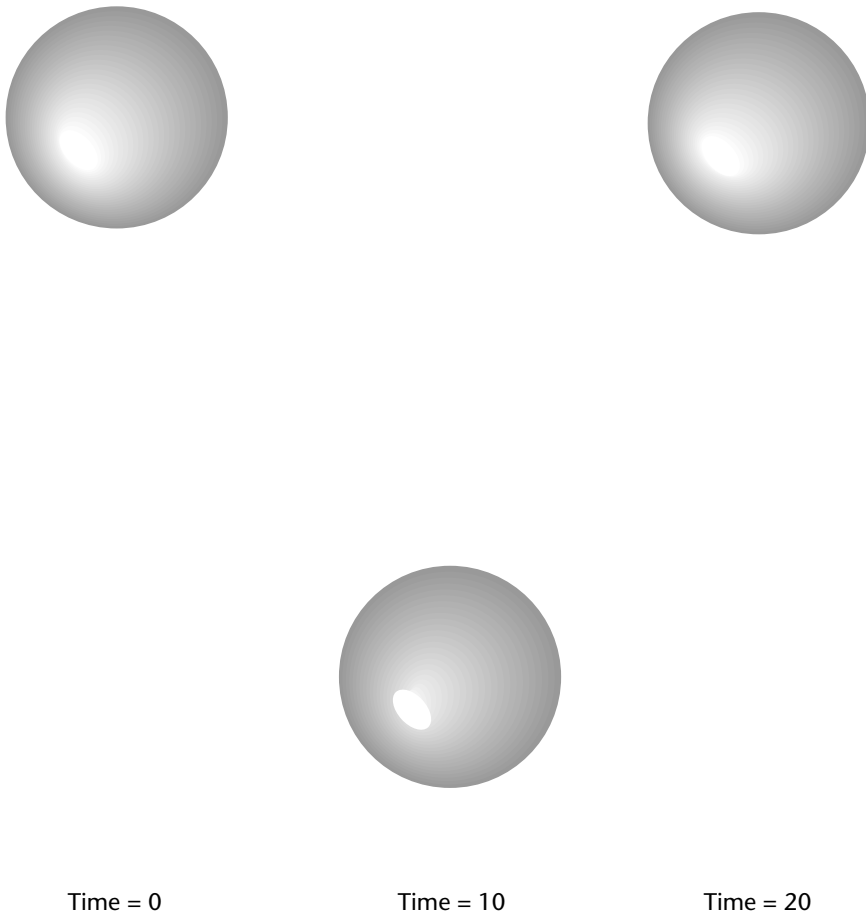


Figure 1.6 Bouncing ball keyframes.

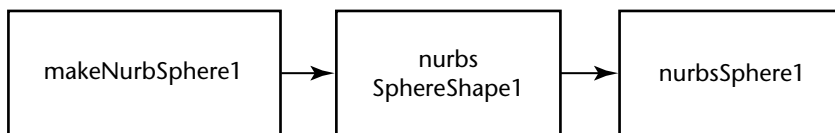


Figure 1.7 Nodes that make up a new sphere.

ber's value to look up. Then, the animation curve sets the `translateY` attribute to the right value. Finally, Maya draws the sphere where you've placed it.

As you can tell, simple operations in Maya, such as making a primitive or animating an attribute, have complex implications under the surface.

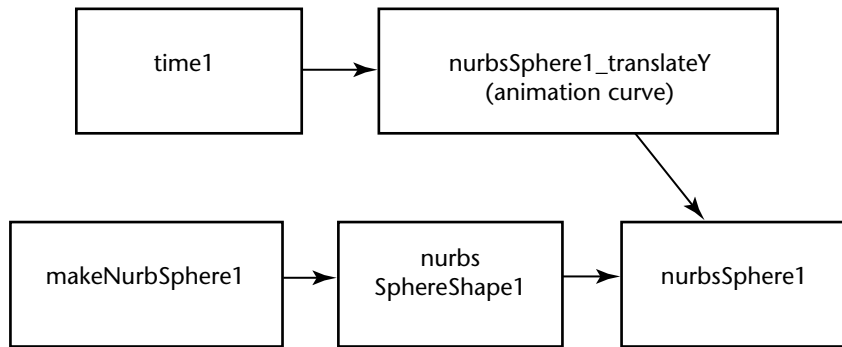


Figure 1.8 Nodes that make up the animated sphere.

One of the strengths of the MEL language is that it can make it easier to set up large networks of nodes to perform a task.

Maya's concept of an expression is more complex than that of a simple connection between attributes, in that an expression can calculate one attribute from another using mathematical operations or almost any other method that you can script in MEL. A connection simply sets one attribute's value to be the same as another's.

How this works is that creating an expression with the Expression Editor or by typing in the Channel Box makes a new node, an expression node. This node contains the expression script and can calculate the expression's output value from its inputs. If you create an expression that calculates an attribute value for node2 from an attribute value in node1, you get the result shown in Figure 1.9. Expressions are discussed much more thoroughly in Chapter 3.

Example 1: Using the Hypergraph to Explore the Dependency Graph

This example demonstrates how to use the Hypergraph to examine how nodes are connected in a scene. First, we'll look at the example, an animated bouncing ball.

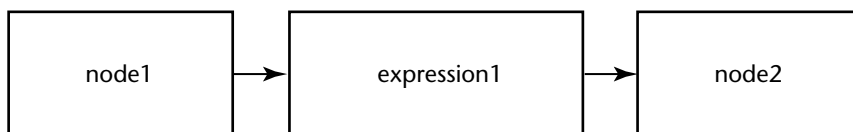


Figure 1.9 An expression node and its connections.

1. Make a new scene by choosing File > New Scene from the main menu.
2. Create a sphere by clicking on the Sphere tool or choosing Create > NURBS Primitives > Sphere.
3. Move the sphere up ten units by typing 10 into the Translate Y field in the Channel Box.
4. Click on the 10 in the Translate Y field of the Channel Box. Then, right-click on the field and choose Key Selected from the menu that appears. The Translate Y field should change color.
5. Click on Frame 10 in the timeline to advance the time. The current time marker should move ahead to Frame 10, leaving behind a red mark indicating a keyframe on Frame 0.
6. Type 0 into the Translate Y field; press Enter to commit the change, and set another keyframe using Key Selected.
7. Click on frame 20 to advance the time, and set a Translate Y keyframe with a value of 10.
8. Rewind and play back the animation.
9. Make sure the sphere is selected, and then choose Window > Hypergraph. . . . When the Hypergraph window appears, use Alt + MMB to drag the view so that the selected sphere object is in the center. Note the trapezoidal shape, which indicates that the sphere has been animated.
10. Choose Graph > Up and Downstream Connections from the Hypergraph menu. Now you should be able to see the nodes that make up the sphere as well as several connected nodes, as shown in Figure 1.10.

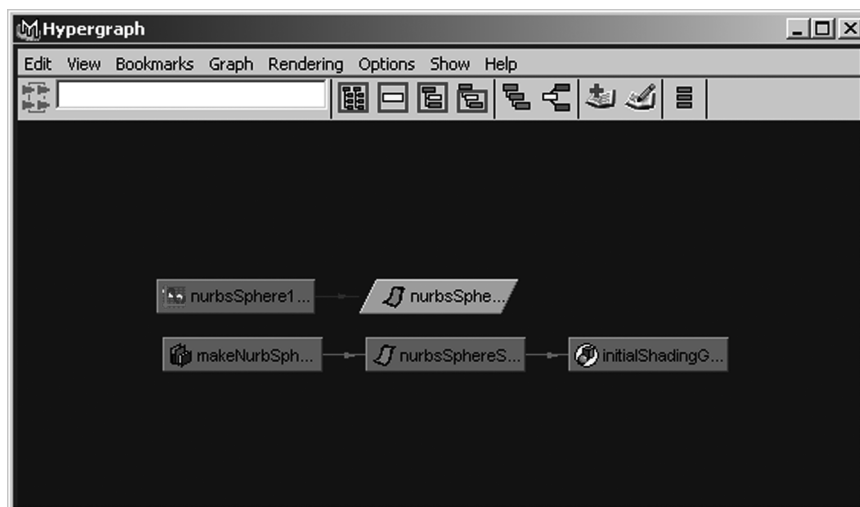


Figure 1.10 Up- and downstream connections for an animated sphere.