

Contents


Tasks

- [Introductory material](#)
 - [Overview](#)
 - [Copyright](#)
- [Getting started](#)
- [Tutorials](#)
- [Opening and saving a scene](#)
 - [Saving objects from a scene](#)
- [Viewing objects and scenes](#)
 - [Examining an object \(rotating, zooming, panning\)](#)
 - [Walking through a scene](#)
 - [Setting a home view](#)
 - [Viewing all models in a scene](#)
 - [Seeking to a point or object](#)
 - [Setting viewer preferences](#)
 - [Defining viewpoints](#)
 - [Global vs. local](#)
- [Importing objects](#)
 - [Importing geometry](#)
 - [Importing an image](#)
 - [Importing a sound](#)
 - [Importing objects as inlines](#)
- [Manipulating objects](#)
 - [Placing shapes when importing, pasting, and creating](#)
 - [Selecting and grouping objects](#)
 - [Creating group hierarchies](#)
 - [Cloning an object](#)
 - [Resizing, moving, and rotating an object](#)
 - [Constraining movement and rotation to a single plane](#)
 - [Aligning two objects](#)

Overview


on this page: [about Cosmo Worlds](#) | [navigating help](#) | [conventions used](#) | [resizing help frames](#)

About Cosmo Worlds

Use Cosmo Worlds to create VRML worlds for publishing on the World Wide Web. Cosmo Worlds fully supports the [Moving Worlds specification for VRML 2.0](#). 

Cosmo Worlds provides you with powerful tools to model complex objects and create exciting, animated worlds. Polygon reduction and other optimization tools ensure that your world will be compact once you publish it on the Web--visitors to your world will find it quick to download and navigate.

Where to go from here:

- To learn about the product's main features by creating a simple world, jump to [Tutorials](#).
- For a complete workflow overview, jump to [Getting Started](#).
- For an overview of the main window, menus, palettes, and toolbar, jump to [A Quick Look at the User Interface](#).
- For tips on how to navigate this help, jump to [Navigating Help](#). 






Navigating Help

Cosmo Worlds Help appears in your Netscape browser window. Click on items in the title bar to navigate the help system:

- *Tasks* and *Reference* make up the *Contents* page. Click on the frame body and use *Ctrl-f* to search for a topic.
- The headings under *Reference* link to overviews of all the tools used for a particular task. The bulleted items link to full descriptions of each tool.

Conventions Used

- [Specifying an object's position in the scene](#)
- [Creating basic shapes and text](#)
 - [Making tube-shaped models](#)
 - [Creating text](#)
- [PEP modeling: editing points, edges, and polygons](#)
 - [Viewing PEP Objects](#)
 - [Selecting PEPs](#)
 - [Translating PEPs](#)
 - [Aligning PEPs](#)
 - [Rotating and scaling PEPs](#)
 - [Deleting, copying, and pasting polygons](#)
 - [Merging, combining, or joining PEPs](#)
 - [Chipping off and forming new polygons](#)
 - [Splitting and cutting polygons](#)
 - [Extruding polygons and edges](#)
 - [Copying polygons and rebuilding models](#)
 - [Mirroring and reordering polygons](#)
- [Changing an object's appearance](#)
 - [Changing the color of a model](#)
 - [Changing the material of an object](#)
 - [Creating your own material](#)
 - [Applying a texture to a model](#)
 - [Editing a texture](#)
 - [Fine-tuning a texture](#)
 - [Creating a new texture palette](#)
 - [Adjusting normals](#)
- [Animating 3D objects](#)
 - [Animating 3D models](#)
 - [Selecting, copying, and pasting keyframes](#)
 - [Animating viewpoints](#)
 - [Try it! Animating a shape](#)
 - [Creating a script:overview](#)
 - [Scripting basics](#)
 - [Steps for creating a script](#)
 - [Try it! Case study for creating a script](#)
 - [Debugging a script](#)
 - [Tips for creating](#)

- Choose *File > Save* means to choose Save from the File pull-down menu.
- *Ctrl-u* means to press the *Ctrl* key and the *u* key at the same time.
- *click* means to quickly press and release the **left** mouse button
- *drag* means to press and hold down the indicated mouse button (the left button if none is indicated)
- *click-middle* means to click the middle mouse button
- *click-right* means to click the right mouse button
- The following [QBullets](#) are used to tell you what you can expect from a link before you click on it:
 - OutLink warns you that the link will take you away from this help system.
 - Scroll Up links to a point further up on the same page.
 - Scroll Down links to a point further down on the same page (except for the "on this page" linked headings).
 - Close Up links to a glossary entry.
 - Picture links to either a standalone image or an HTML page dominated by a single picture.

Resizing Help Frames

By default, *Cosmo Worlds Help* consists of three frames: the title bar, the side bar with contents, and the body. You can resize any of these frames by dragging the frame divider. When you resize the whole Netscape window, the frames reset themselves to their original sizes--you will have to resize the frames again.

For a frames lite version of the help, click the *SmallHelp* button on the title bar. This replaces the three-frame help with a larger window and small navigation bar. Note that switching to SmallHelp replaces the text in the main window with the Contents.

Tip: When using SmallHelp, you can free up more window space by hiding Netscape's Toolbar, Location, and Directory Buttons. From Netscape's menu bar, deselect any of the following: *Options > Show Toolbar, Show Location, Show Directory Buttons*.

Some stand-alone diagrams appear too large to fit in your Netscape window. These are large .gif images not embedded in an HTML page. You can resize the window by dragging its edges or edit your *general preferences* for helper applications to view these images in their own window. See Netscape's Handbook for more details.

[© Copyright 1996, Silicon Graphics, Inc. - All Rights Reserved](#)

- [scripts](#)
 - [Advanced: creating a script from scratch](#)
 - [Creating a switch](#)
 - [Creating a trigger](#)
- [Adding special objects to a scene](#)
 - [Adding navigation information](#)
 - [Adding lights](#)
 - [Tips for efficient lighting](#)
 - [Light types: what's the difference?](#)
 - [More about spotlights](#)
 - [Try It! Experimenting with lighting effects](#)
 - [Adding sounds](#)
 - [Try it! Creating an ambient sound](#)
 - [Try it! Creating an event sound](#)
 - [Recording sounds](#)
 - [Adding links \(anchors\)](#)
 - [Adding billboards](#)
- [Using the outline editor](#)
 - [Creating routes](#)
 - [Cloned nodes in scripts and prototypes](#)
- [Optimizing the scene](#)
 - [Checking the polygon count](#)
 - [Reducing polygon count](#)
 - [Building efficient polygons](#)
 - [Using inlined objects](#)
 - [Using multiple levels of detail](#)
 - [Controlling collision detection](#)
- [Packaging the scene](#)
 - [Steps for packaging](#)
 - [Try it! Packaging a sample world](#)
 - [Links followed during discovery](#)
 - [Possible packaging errors](#)
- [Customizing the environment](#)
 - [Adding a helper application](#)
 - [Helper application database file](#)
 - [Changing the background color](#)

Reference

The Cosmo Worlds Window

- [A quick look at the user interface](#)
(menus and palettes)
- [Keys and shortcuts](#)
- [Viewers](#)
 - [Examiner Viewer](#)
 - [Walk Viewer](#)
 - [Plane Viewer](#)
 - [Viewer Popup Menu](#)
 - [Viewer Menu Preferences](#)

Cosmo Worlds Tools and Utilities

- Creators
 - [Text Editor](#)
 - [Extrusion Editor](#)
- Editors
 - [Outline Editor](#)
 - [NavigationInfo Editor](#)
 - [Sound Editor](#)
 - [Viewpoint Editor](#)
 - [Polygon Copier](#)
 - [Polygon Reduction Editor](#)
- [Media Tools](#)

Contents depends on system configuration; possibilities include:

 - ImageWorks Plug-in Editor
 - Alias/Wavefront Plug-in Editor
 - Photoshop Plug-in Editor
- Action Tools
 - [Script Editor](#)
 - [Keyframe Animator](#)
 - [Link Editor](#)
 - [Inline Editor](#)
 - [Switch Editor](#)
 - [Level of Detail Editor](#)
 - [Billboard Editor](#)
 - [Collision Editor](#)
- [Layout Tools](#)
 - [Snap Target and Snap Source](#)
 - [Precision placement panel](#)
 - [PEP Alignment](#)
 - [PEP distance snap options](#)
 - [PEP angle snap option](#)
- [PEP Editing Tools](#)
 - [Selection](#)
 - [PEP Editor](#)
 - [Convert to PEP](#)
 - [PEP Rotation / Scaling](#)

- [Cut Polygons](#)
 - [Merge PEP Objects](#)
 - [Join Selected Polygons](#)
 - [Chip Off and Split Selected Polygons](#)
 - [Extrude Selected Polygons or Edges](#)
- Looks Tools
 - [Material Palette](#)
 - [Normal Doctor](#)
 - [Material Editor](#)
 - [Texture Editor](#)
 - [Color Per Vertex Editor](#)
 - [PEP Texture Applicator](#)
 - [Light Editor](#)
- [CosmoPackage](#)
 - [Root documents](#)
 - [VRML preferences](#)
 - [Directory index](#)
 - [Mappings](#)
 - [Trusted references](#)
 - [Inspecting documents](#)
- [Glossary](#)

Overview

on this page: [about Cosmo Worlds](#) | [navigating help](#) | [conventions used](#) | [resizing help frames](#)

About Cosmo Worlds

Use Cosmo Worlds to create VRML worlds for publishing on the World Wide Web. Cosmo Worlds fully supports the [Moving Worlds specification for VRML 2.0](#).🌐

Cosmo Worlds provides you with powerful tools to model complex objects and create exciting, animated worlds. Polygon reduction and other optimization tools ensure that your world will be compact once you publish it on the Web--visitors to your world will find it quick to download and navigate.

Where to go from here:

- To learn about the product's main features by creating a simple world, jump to [Tutorials](#).
- For a complete workflow overview, jump to [Getting Started](#).
- For an overview of the main window, menus, palettes, and toolbar, jump to [A Quick Look at the User Interface](#).
- For tips on how to navigate this help, jump to [Navigating Help](#).🔗






Navigating Help

Cosmo Worlds Help appears in your Netscape browser window. Click on items in the title bar to navigate the help system:

- *Tasks* and *Reference* make up the *Contents* page. Click on the frame body and use *Ctrl-f* to search for a topic.
- The headings under *Reference* link to overviews of all the tools used for a particular task. The bulleted items link to full descriptions of each tool.

Conventions Used

- Choose *File > Save* means to choose Save from the File pull-down menu.
- *Ctrl-u* means to press the *Ctrl* key and the *u* key at the same time.

- *click* means to quickly press and release the **left** mouse button
- *drag* means to press and hold down the indicated mouse button (the left button if none is indicated)
- *click-middle* means to click the middle mouse button
- *click-right* means to click the right mouse button
- The following [QBullets](#) are used to tell you what you can expect from a link before you click on it:
 - OutLink warns you that the link will take you away from this help system.
 - Scroll Up links to a point further up on the same page.
 - Scroll Down links to a point further down on the same page (except for the "on this page" linked headings).
 - Close Up links to a glossary entry.
 - Picture links to either a standalone image or an HTML page dominated by a single picture.

Resizing Help Frames

By default, *Cosmo Worlds Help* consists of three frames: the title bar, the side bar with contents, and the body. You can resize any of these frames by dragging the frame divider. When you resize the whole Netscape window, the frames reset themselves to their original sizes--you will have to resize the frames again.

For a frames lite version of the help, click the *SmallHelp* button on the title bar. This replaces the three-frame help with a larger window and small navigation bar. Note that switching to SmallHelp replaces the text in the main window with the Contents.

Tip: When using SmallHelp, you can free up more window space by hiding Netscape's Toolbar, Location, and Directory Buttons. From Netscape's menu bar, deselect any of the following: *Options > Show Toolbar, Show Location, Show Directory Buttons*.

Some stand-alone diagrams appear too large to fit in your Netscape window. These are large *.gif* images not embedded in an HTML page. You can resize the window by dragging its edges or edit your *general preferences* for helper applications to view these images in their own window. See Netscape's Handbook for more details.

[© Copyright 1996, Silicon Graphics, Inc. - All Rights Reserved](#)

Tutorials

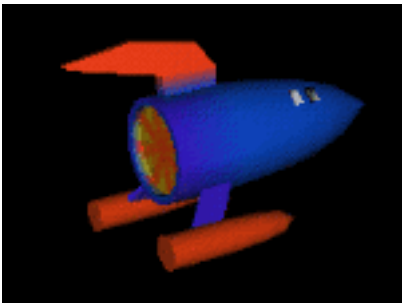
[Creating a House](#)

Learn how to model a simple cube into a house by editing its points, edges, and polygons. This short tutorial focuses on basic modeling technique.



[Creating and Animating a Rocket](#)

Learn about Cosmo Worlds' most powerful features in this four-part tutorial. By creating a planet, assembling and animating a rocket, and previewing the final product, you step through a typical workflow scenario and use Cosmo Worlds' most common scene-creation tools.



Try It!

The following mini-tutorials showcase an editor or tool:

[Try It! Animating a Shape](#)

[Try It! Experimenting with Lighting Effects](#)

[Try It! Creating an Event Sound](#)

[Try It! Creating a Script](#)

[Try It! Packaging a Sample World](#)

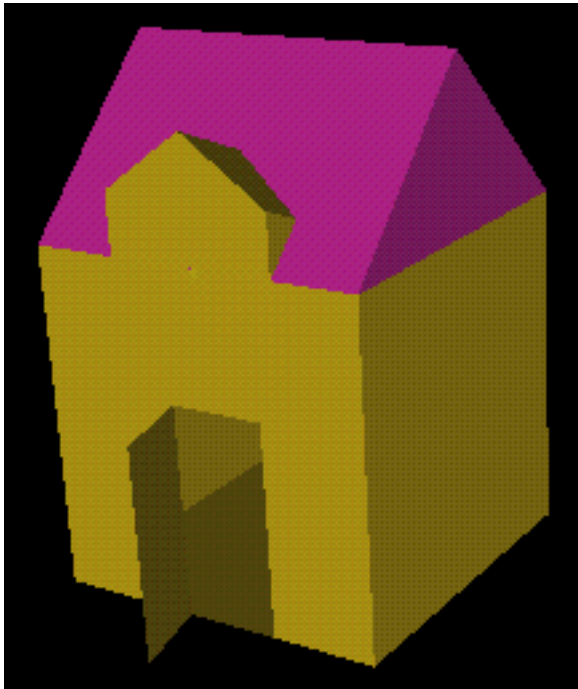
Creating a House

Level: Beginner

Goal: Learn basic skills for PEP modeling (editing points, edges, and polygons). Also learn how to change the appearance of a model by coloring vertices and specifying backface culling.

Assumptions: You should be familiar with the concept of PEP modeling. See [PEP Modeling: Editing Points, Edges, and Polygons](#) in the online help for more details. You should also be familiar with the Cosmo Worlds user interface. See [A Quick Look at the User Interface](#) for more details.

Finished product:



This tutorial consists of the following sections:

- [Creating a Roof for a House](#)
- [Creating a Dormer for a House](#)
- [Creating and Opening a Door](#)
- [Coloring the House](#)
- [Looking Inside the House](#)

Use Undo and Redo to fix mistakes:

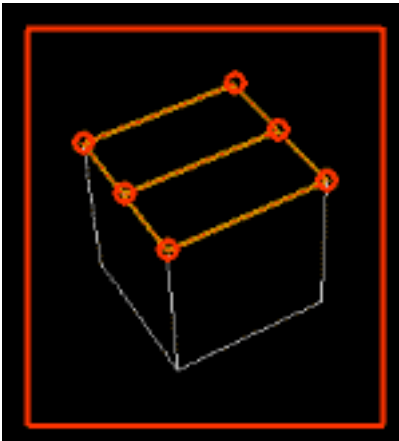
- Undo = *Ctrl*+Z
- Redo = *Shift*+*Ctrl*+Z

Creating a Roof for a House

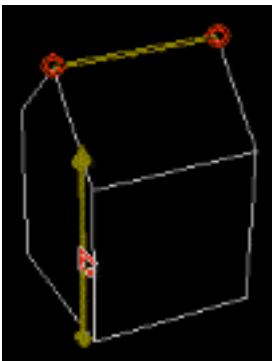
The basic shape for a house is fairly simple.

1. Start out with a cube in the PEP editor. Switch to hidden line so you can see the individual polygons easily (press and hold the right mouse button over your workarea and choose *Draw Style > Hidden Line* from the popup menu).

2. Select the top polygon and press  to split the top polygon lengthwise.



3. Select the edge in the middle of the split polygon and *Shift*-drag along a side polygon to push the edge up and form the roof:




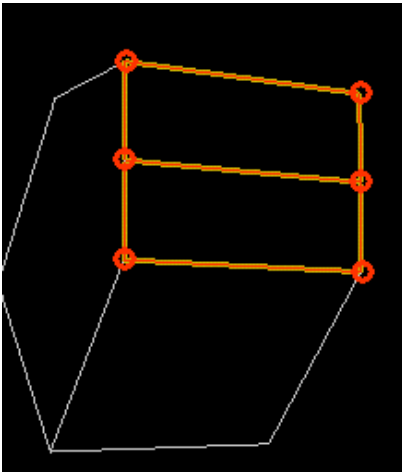
Creating a Dormer for a House


Use Split Buttons to Define the Dormer

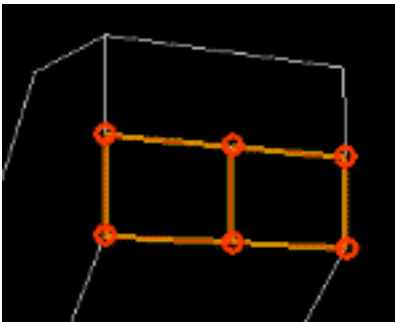
You can model interesting shapes from a simple PEP object like a square by using a combination of split and cut operations, then extruding and moving the new polygons. Here's how to create a dormer from the basic house shape.




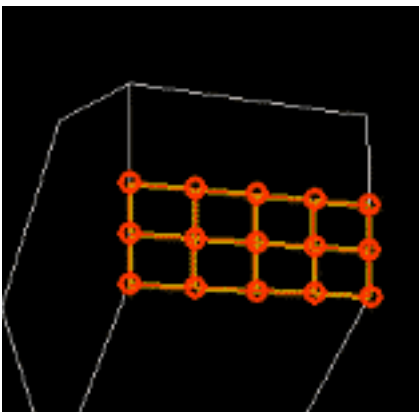
1. Starting with the basic house shape in Example 1, select a side of the roof and press  to split the polygon lengthwise:





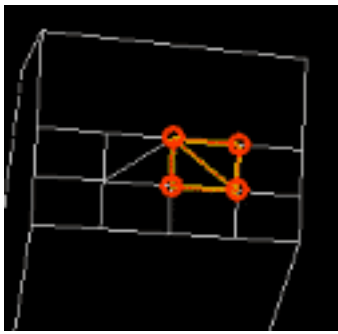
2. Select the lower half of the roof and press  to split the polygon vertically:



3. Keeping the same selection from step 2 (don't select anything new), press  to split the quads like this:



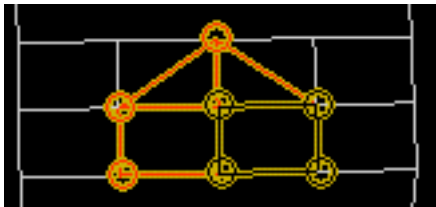
4. To make the angles for the dormer roof, select the appropriate quad and press  or :




Now you have the basic shape for the dormer.

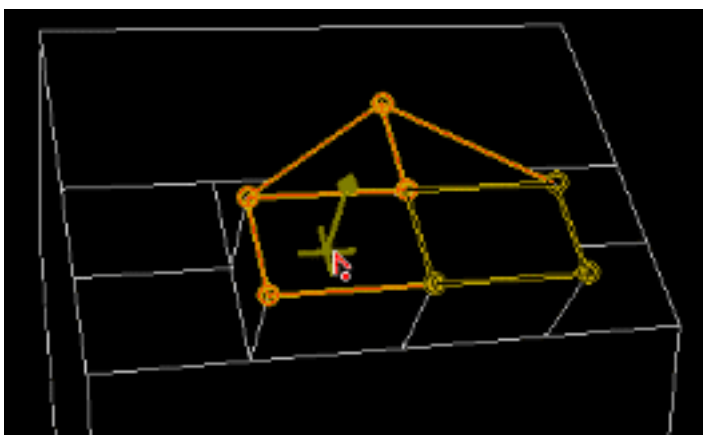
Extrude, Rotate, and Pull Out the Dormer

1. To pull out the dormer, *Shift*-click to select the four polygons that make up the dormer. The last polygon you click becomes the master selection and is outlined in black:



The master selection is used for aligning the front face of the dormer to the front face of the house. If you lose the master selection in later steps, you can reselect it by *Shift*-clicking the polygon twice--the first time deselects the polygon and the second time adds it to the selection and specifies it as the master selection.

2. Move the dormer up and away from the rest of the roof. Make sure you have the correct polygons selected as described in step 1. Then press the Extrude Polygons button . Place the mouse cursor over the dormer and *Ctrl*-drag out, so the dormer moves away from the house and its edges are perpendicular to the roof.




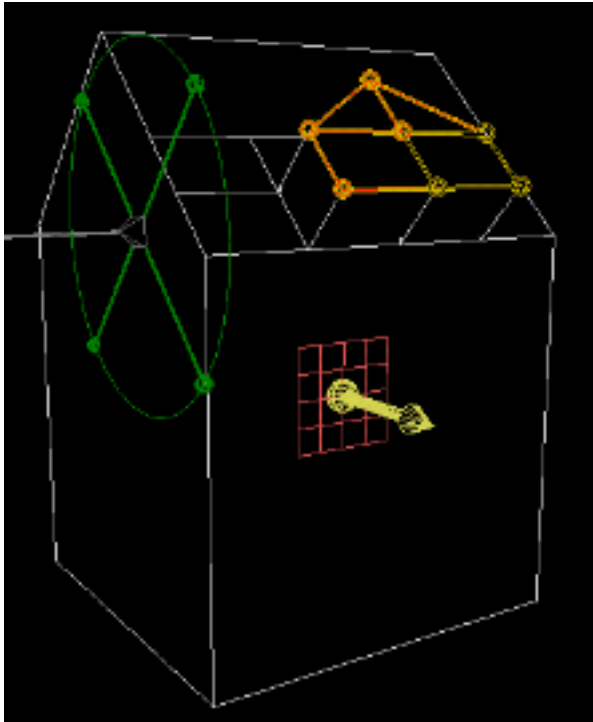
Next you want the dormer's face parallel to the front of the house. To fix the angle, you'll use the Snap Target and PEP Jack.

3. Place the Snap Target by placing the mouse cursor on the front of the house and clicking the

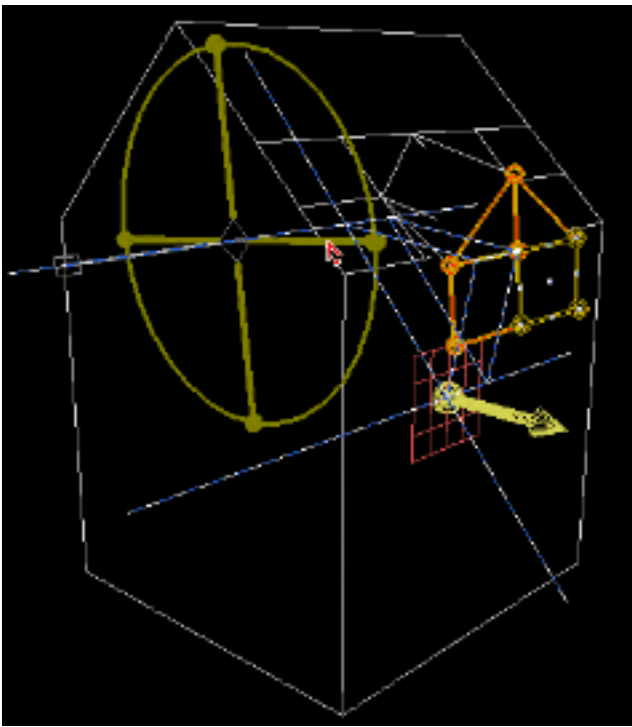
middle mouse button. A yellow arrow appears (see image in step 4). The target is used to align the master selection (specified in step 1) to the front of the house.



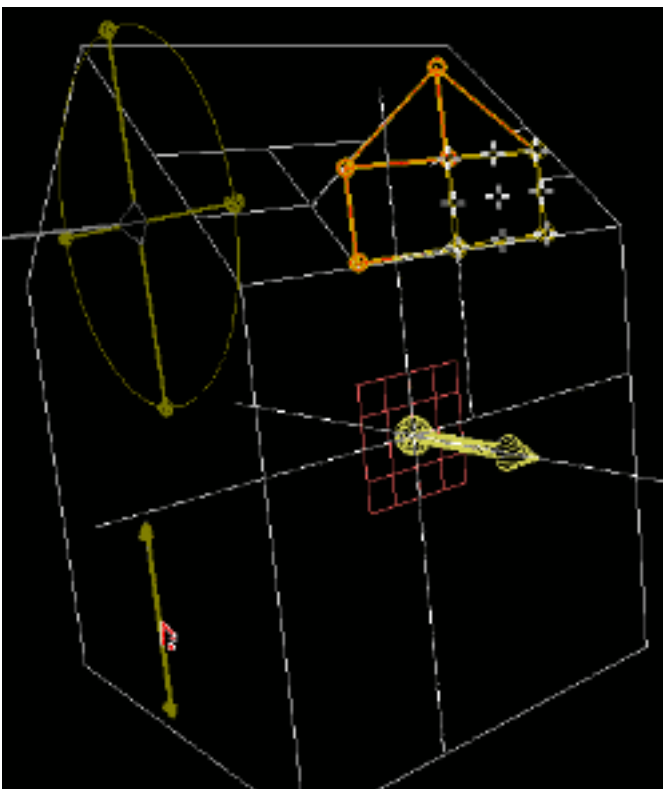
4. Click the PEP Jack button  and place the tool by clicking in the scene. Position the tool by dragging it. The PEP Jack doesn't need to be on the master selection or near it, but it does need to be positioned so the ring and green balls are in the same plane as the side of the house (see image below).




5. Make sure the dormer polygons are selected and the master selection is specified (see step 1). Rotate the PEP Jack by dragging a green knob. The PEP Jack pulls the dormer face down. Drag the PEP Jack until the front of the dormer is parallel to the front of the house. Dotted lines indicate the master selection's alignment relative to the Snap Target; use this feedback to help you position the dormer correctly.



6. Before the dormer is complete, you need to move it up so that it looks right. First, make sure the polygons making up the dormer are selected. Position your cursor over the front of the house, and *Shift*-drag up to move the dormer so that the bottom edges are aligned with the top edge of the polygon making up the front of the house. Again, use the feedback from the Snap Target to help you position the dormer correctly. Starbursts appear along the master selection to indicate its alignment.



7. Take a moment to clean up your workarea; put away the PEP Jack  and the Snap Target



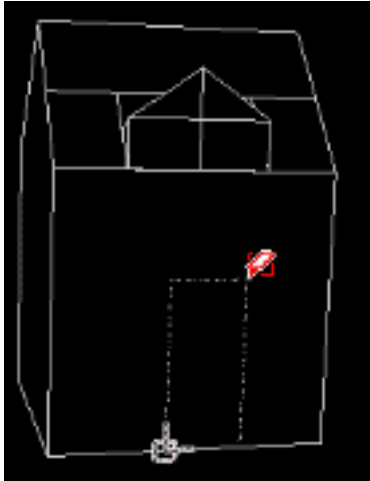
by clicking their buttons on the PEP and Layout palettes.

Creating and Opening a Door

Now that the dormer is correct, you want to create a door for the house.



1. Click the Cut Rectangle button (on the PEP palette) and drag the cutting tool across the front of the house to define a new polygon for the door:



2. If you try to select and move this polygon, it will pull out the points that define the front of the house. To move just the polygon that defines the door, select it and click the Chip Off Polygon

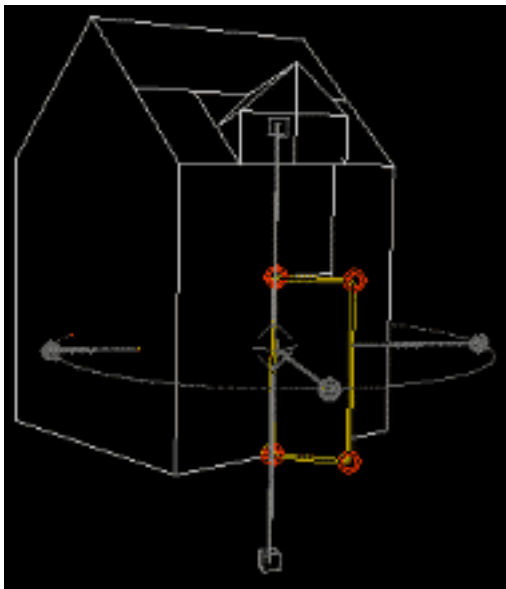


button.

3. The result is a polygon you can move independent from the rest of the house.



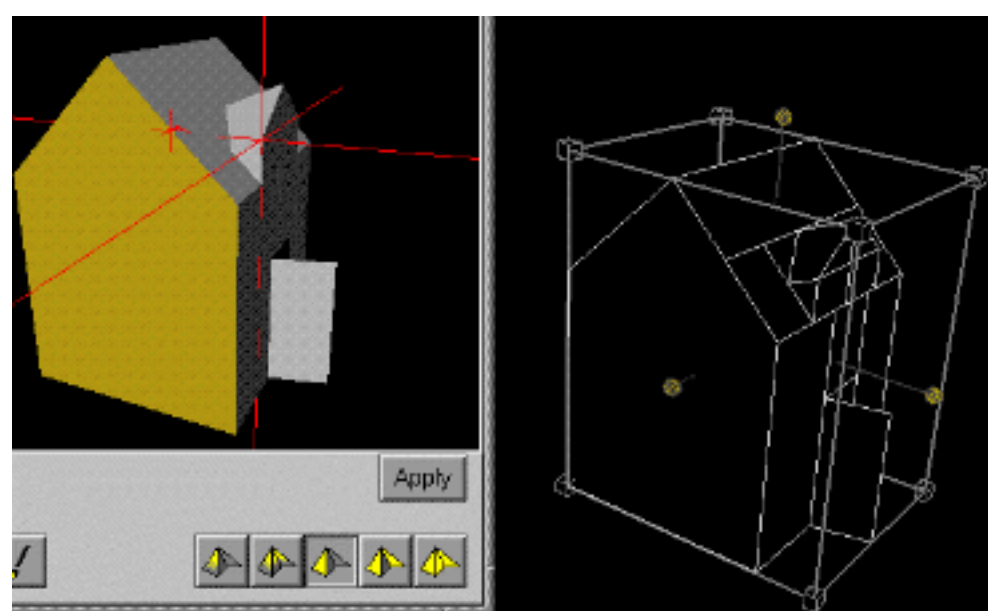
4. Next, rotate the edge of the door like a hinge. Place the PEP Jack in the scene and align it along the edge of the door as shown below:



Make sure the polygon making up the door is selected, then drag a green knob on the PEP Jack to rotate the door open.

Coloring the House

To color the house, use the Color Per Vertex Editor. This editor is pretty easy to use (for the basics, see the [Color Per Vertex Editor](#) quick reference). However, there's a trick to get the color of the roof separate from the rest of the house. Right now, each roof's side is made up of a single polygon that makes up the whole side of the house. When you try to color the roof, here's what happens:

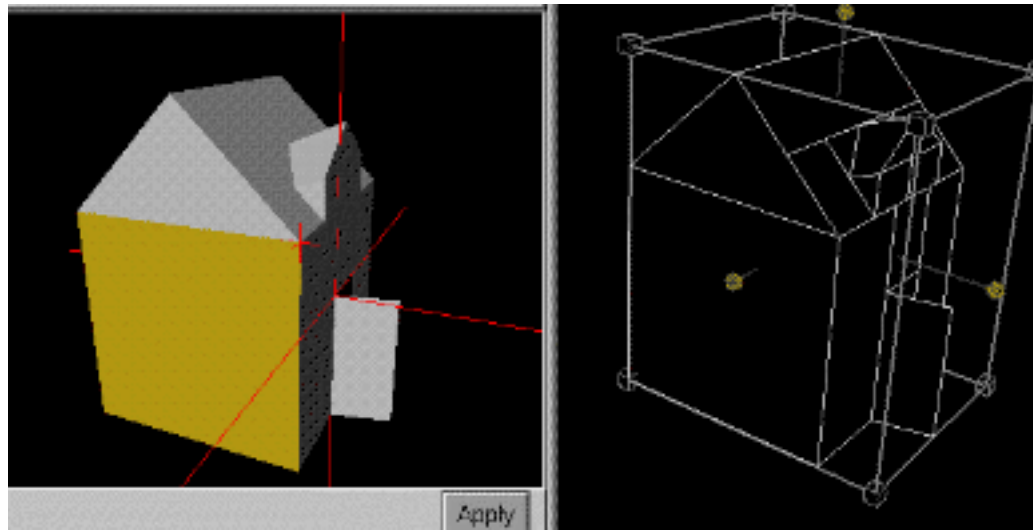


Note: This image shows part of the Color Per Vertex editor next to the workarea. The model in the workarea is shown in Hidden Line draw style.

To color just the bottom square making up the side of the house, you first need to define the side

polygon as two separate polygons. Click the Cut Polygon button  on the Split palette. Click to define the first endpoint for the new edge and drag out a line, releasing the mouse button where you

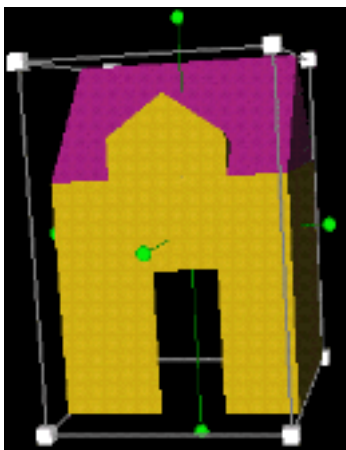
want to create the second endpoint. You now have a new edge that separates the side polygon into two polygons, separating the roof from the rest of the side of the house. Repeat this process to separate the roof on the other side of the house. When you color a side of the house, it should now look like this:



Looking Inside the House

Now that your house is colored, switch from Hidden Line draw style to As Is. (Choose *Draw Style* > *As Is* from the right mouse popup menu.) Notice that when you look inside your house, you can't see the inside of the walls or floor. To fix this, open the Normal Doctor and deselect the box next to Cull Backfacing Triangles.

1. Select the house (you should not be in PEP Editing mode).

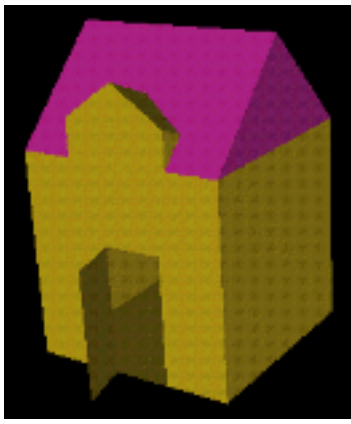


2. Click the Normal Doctor button on the Looks palette:



3. Click the box next to Cull Backfacing Triangles to remove the checkmark.

Now you can see the walls and floor inside the house.



That's it! For more information on any of these topics or tools, refer to the appropriate section in the online help.

PEP Modeling: Editing Points, Edges, and Polygons

on this page: [the basics](#) | [task summary](#)

PEP modeling lets you edit the individual points, edges, and polygons of objects in your scene. Used in conjunction with optimization editors like the Polygon Reducer, PEP modeling can be a powerful way to build models that appear complex, yet maintain a low polygon count.

The Basics

- **PEP modeling** means you are editing an object's points, edges, and polygons.
- **Palettes**--PEP modeling has two palettes called *PEP* and *Split*. You press a button to execute or set up most modeling tasks.
- **Convert to PEP Object**--PEP modeling works only with PEP objects. Any object in your scene can be changed to a PEP object by selecting it and pressing this button:



A PEP object is a shape in your scene that has been converted to an [IndexedFaceSet](#). 🔍

- **PEP Editor**--PEP Modeling takes place within a special mode referred to as the *PEP Editor*. To enter this mode, select a PEP object and press *Ctrl-e*, or click the *Select PEPs* button:



An orange box appears around the PEP object and the arrow cursor changes to indicate that you are now in PEP editing mode. To exit the PEP Editor, click anywhere outside the orange box in your scene, or press the Select PEPs button again.

- **Infinite Undo and Redo**--*Ctrl-z* to undo, *Shift-Ctrl-z* to redo. You can undo and redo up to the last time you saved the file or until you run out of memory.

Task Summary

Here are the basic steps to PEP modeling:

1. Select an object in your scene by clicking on it.
2. Turn the object into a PEP object: select the object, then press the *Convert to PEP object* button:



This button will be grayed-out if your object is already a PEP object.

3. Enter the *PEP editor*. Click on the PEP object to select it and press *Ctrl-e*, or click the *Select PEPs* button:



4. [Position your view](#) of the object.
5. [Select individual or multiple points, edges, and polygons](#).
6. [Translate PEPs by dragging](#), or [use the layout tools to align PEPs](#).
7. Edit the PEP object:
 - [Rotate and scale PEPs with the PEP Jack](#)
 - [Delete, copy, and paste polygons](#)
 - [Extrude polygons and edges](#)
 - [Merge, combine, or join PEPs](#)
 - [Chip-off polygons or form a new polygon from selected edges](#)
 - [Split and cut polygons](#)
8. Once you edit the model, use the [Normal Doctor](#) to fix normals, crease angles, or backface culling.
9. Optimize the model by [reducing the number of polygons](#), or use the [polygon copier](#) to break down and rebuild polygons.

Jump to: [Tools to Use: Creating and Editing Models](#)

Expand Your Vocabulary

avatar

a physical manifestation of the user at the current viewpoint in a world.

backface culling

specifies that polygons not facing the camera are not rendered (if you can't see the polygons, don't waste time rendering them). This is useful for solid objects.

billboard

an object that rotates around a specified axis so that it always faces the viewer. Billboards are a useful optimization technique.

browser

a software program that can view VRML worlds. Also called a *viewer* or a *player*.

clipping plane

clipping planes define the area in the scene visible to the camera. The near clipping plane is the point closest to the camera that is still visible; the far clipping plane is the point farthest from the camera that is still visible.

clone

a copy, or *instance*, of an object. A clone defines the selected object as a master for producing multiple instances. All instances share this master description identically, tracking all changes to its shape and appearance.

crease angle

the angle between the normals for two adjacent faces of a polygon.

dihedral angle

the angle between two adjacent polygons.

edge

where two faces (polygons) of a model meet. The term *edge* is different than the term *line*, in that it implies membership in the *face*.

event

an indication that something has happened. Outgoing events send their values to incoming events, which receive values. The connection between two events is called a *route*.

face

a single polygon, defined by points and lines; may also refer to a "perceived" face, which is a rendered "side" of an object that may be composed of several polygons. A group of faces make up a model's surface.

iconic objects

geometric placeholders for nodes which will be invisible to a VRML browser, for example,

sound and light nodes.

IndexedFaceSet

a VRML node that defines the coordinates for a set of polygons that make a shape.

inline

a file referenced by URL (Uniform Resource Locator) within another file. In VRML files, inlines are downloaded separately from the main file and are a good way to speed up transmission and rendering of the main file. Some browsers display inlines as bounding boxes while they are loading.

keyframe

a pose at a particular time. The Keyframe Animator interpolates between poses, filling in values to move from one pose to the next. A keyframe can also contain values for viewpoints, textures, materials, and shapes. The Keyframe Animator interpolates between the values to transition from one viewpoint, texture, material, or shape to the next.

line

defined by two points (vertices) of a model; also see *edge*.

LOD (Level of Detail)

an object which appears to change to display the appropriate level of detail depending on viewing distance. LODs are, in fact, multiple representations of the same object.

For example, if you are looking at a forest from far away, you don't need to see the individual trees. The LOD might be made of large polygons that represent a forest. As you get closer, the LOD is replaced by another LOD made with more polygons so that you can see tree trunks and leaves.

mapping

instruction for placement of packaged files.

member

an object within an animation that moves as a unit in the scene.

normal vector (or normal)

a directional line that is perpendicular to a surface. A normal indicates the front of a face.

object

a single 3D item in a VRML scene, or an item that is part of a grouped object's hierarchy.

orthographic projection

2D view of objects in a scene; typical orthographic views are top, front, and side.

package

to locate all the files necessary to create a document or world and organize them for publication on a Web server.

PEP

Points, Edges, Polygons. Refers to Cosmo Worlds's interface to the VRML node *IndexedFaceSet*.

PEP Editor

the mode used to model PEP objects (PEP modeling). The PEP Editor is indicated by an orange frame around the object you are editing.

PEP object

an object which has been converted to its individual points, edges, and polygons.

PEP modeling

editing an object's points, edges, and polygons; edits the coordinates inside the *IndexedFaceSet* node.

perspective projection

3D view of objects in a scene.

point

a single vertex in a model.

polygon

in Cosmo Worlds, a face of a model, defined by the model's points and edges.

proxy

an object to be used for collision detection in place of actual geometry, which is usually more complex than the proxy.

publish

to copy a set of packaged files to a Web server where they can be accessed by Web clients.

root

a top-level document or world in the packaging directory. All files referenced in this document, as well as the local files referenced by those files, will be included as part of the package.

route

connection between two events.

surface

a set of faces; the surface of a cube includes all of its faces.

VAG (VRML Architecture Group)

a group of technical experts formed to set standards for creating and displaying 3D worlds on the Web. See the [VAG home page](#) for additional information.

VRBS (Virtual Reality Behavior System)

a VRML behavior scripting system prototype developed by the San Diego Supercomputer

Center (SDSC). See the [VRML Behaviors home page](#) at the SDSC for more information.

VRML (Virtual Reality Modeling Language)

VRML is a standard way to specify 3D objects for viewing across the World Wide Web.

VrmlScript

a programming language, similar to JavaScript, that is used for scripts contained in a Script node. [Click here](#) for the complete VrmlScript specification.

Viewing Objects in the PEP Editor

Use the [Examiner Viewer](#) and [Walk Viewer](#) controls to reposition the camera around an object in the PEP Editor. If you haven't already done so, read [Viewing Objects and Scenes](#) to learn about this topic. If you are already familiar with this topic, you can jump to [viewing shortcuts](#).

Jump to:

- [PEP Modeling](#)
- [Tools to Use: Creating and Editing Objects](#)

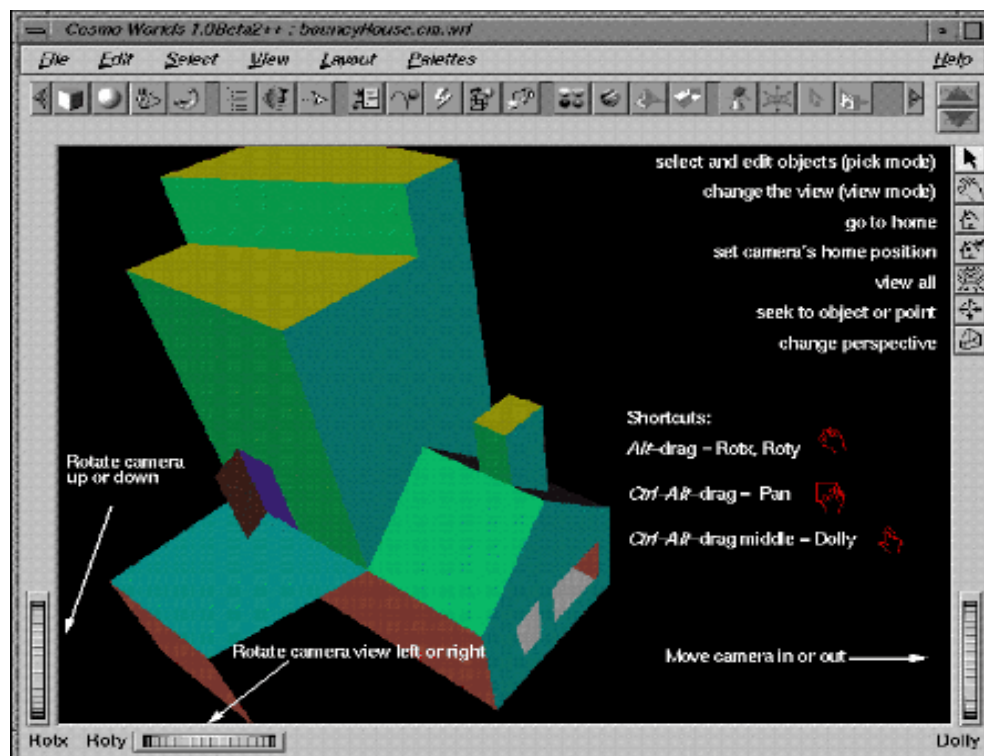
Examiner Viewer

on this page: [how it works](#) | [viewer controls and menus](#) | [shortcuts](#)

Use to: Rotate, dolly, and pan when viewing objects

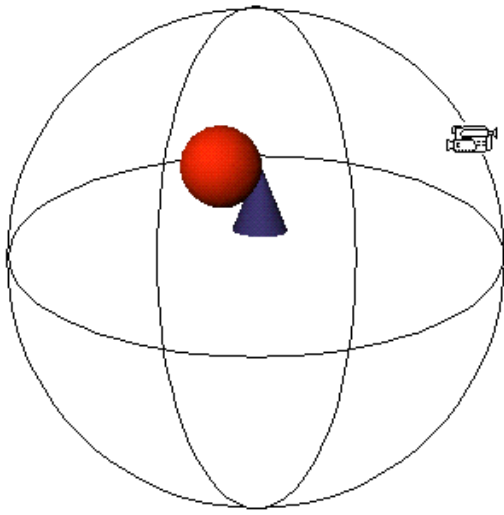
Find it: Choose *View > Examiner Viewer*, or *Shift-Ctrl-e*

How It Works



[Click image for full view.](#)

The Examiner Viewer uses a virtual sphere metaphor--the scene is contained inside a sphere that the camera moves around:



As a result, it seems like the Examiner Viewer treats all the objects in your scene as a single, grouped object.

Tip: If you have many objects in your scene and have trouble viewing a single object that you are editing, select that object and choose *View > Hide All Unselected*. You also might want to [save the object into its own Cosmo Worlds file](#), edit it, and reimport it as an inline.

The Examiner Viewer is one of two viewers in Cosmo Worlds. The other viewer, called the [Walk Viewer](#), lets you navigate through a scene by moving the camera using a walking or driving metaphor.

Both the Examiner Viewer and the Walk Viewer have a menu that you access by holding your right mouse button over the viewing area. Use this [Viewer Menu](#) to [change the draw style](#) (rendering characteristics), and [preferences](#).


Tricks:

- You can spin the camera around an object with the hand icon (*Alt*-drag) by dragging and letting go. Release *Alt* to stop the spin.



- Click the *Seek* button, then click an object or point in your scene to move quickly toward it. Click again before the movement stops to seek toward a different object or point.

Viewer Controls and Menus

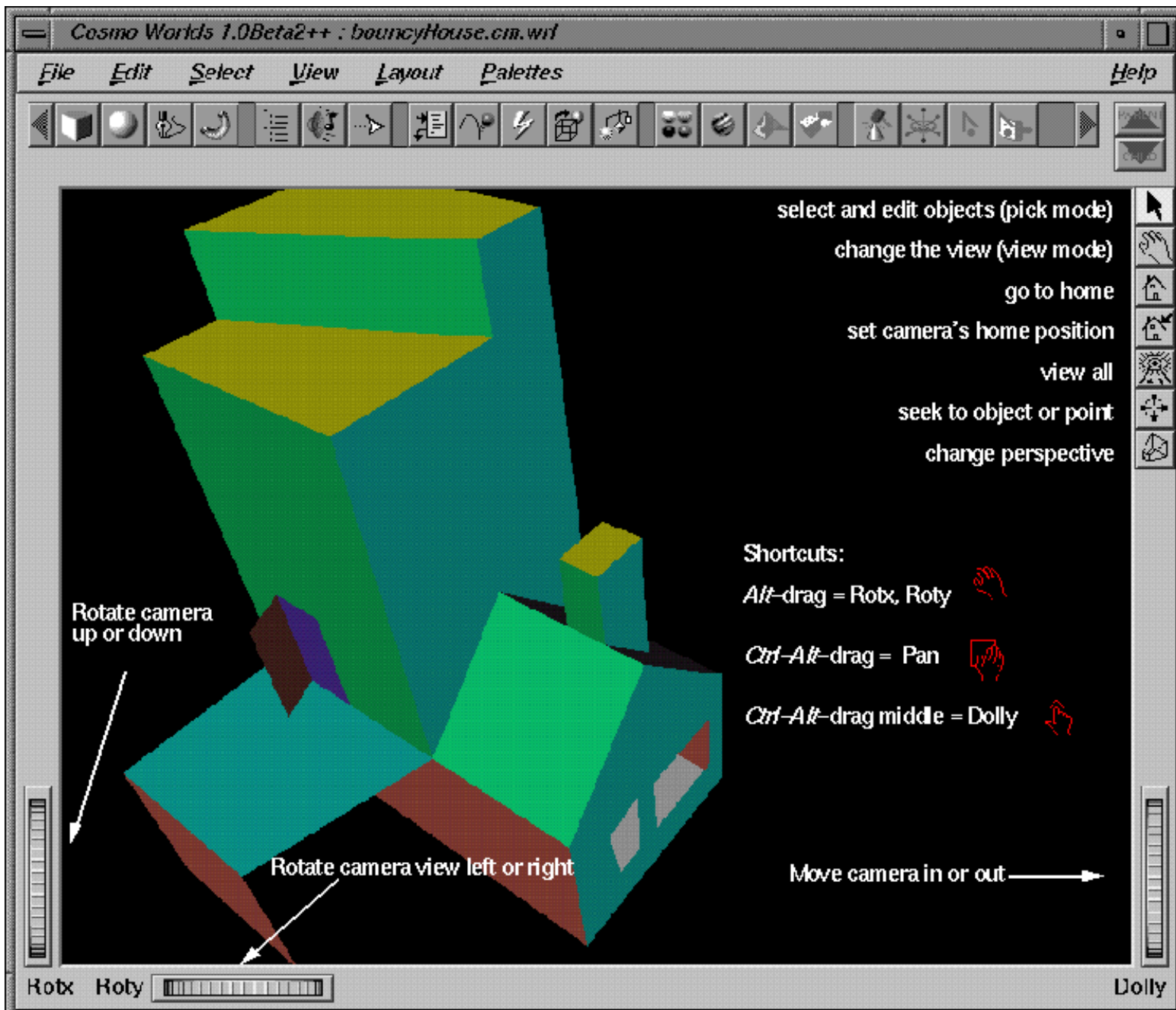
If you haven't looked at the [Examiner Viewer diagram](#), do so now. This image briefly describes the Examiner Viewer's controls and toolbar buttons. For a full description of the viewer toolbar and the View palette, see [Tools to Use: Viewing](#). There is also a viewer menu (accessed with the right mouse button). Use this to change the draw style and viewer-related preferences. See [Viewer Menu](#) for details.

Shortcuts

You can use keyboard shortcuts instead of the viewer controls. See [Keys & Shortcuts for Viewing](#).

Jump to:

- [Viewing Objects and Scenes](#)
- [Tools to Use: Viewing](#)
- [Keys & Shortcuts for Viewing](#)



Saving Objects from a Scene

If you want to rework an an object you have created within a larger scene, it can be easier to save that object into its own file, edit it and import the file into your scene.

To save an object in your scene into its own file:

One way is to save the scene as a new file with a different filename then delete all the other objects in your original scene and save the new scene (containing the isolated object) with a new name.

Another way is to select and cut the object you want to isolate (choose *Edit > Cut*), save the file before closing so that it doesn't include the isolated object, open a new file (choose *File > New*), paste the isolated object into the new scene (choose *Edit > Paste*), and save the file with a new name.

Once you have finished working on the object, you can cut and paste it into the original scene, [import it](#) (*File > Import*) , or [import it as an inline](#) (*File > Import As Inline*).

Jump to:

- [Opening and Saving a Scene](#)
- [Importing 3D Objects](#)
- [Importing Inlines](#)
- [Creating and Editing Models](#)

Importing 3D Objects

on this page: [supported file formats](#) | [interactive file placement](#)

Find it: Choose *File > Import*

(or drag and drop the file icon from the desktop into the main window)

You can import 3D objects into your scene using a variety of methods. Cosmo Worlds supports a number of file formats, so you can take advantage of 3D clip-art libraries, and you can reuse objects created using other modeling packages. If you use the drag-and-drop method to import 3D objects, you can select multiple files and drag them all into the main window at the same time.

Check out the directory `/usr/share/data/vrml` for a variety of VRML clip-art files.

Supported File Formats

You can import the following file formats into a Cosmo Worlds scene:

- VRML 1.0
- VRML 2.0
- Inventor

If you install *xlators_3d* from the IRIX 6.*n* CD, you can also import:

- Alias
- DXF
- Obj
- SLA
- Softimage

Interactive File Placement

By default, you can control where an imported object is originally placed in the scene. Turn off interactive import placement using the check box in the *File* menu. Here are the details of how it works:

- If the snapping target is active when the imported object is placed in the scene, the object is always placed at the snapping target (regardless of what the check box says).
- If the snapping target is not active, interactive placement is ON (the box in the *File* menu is checked), and the scene contains at least one object, you can drag the imported object around in the scene until you're satisfied with its placement. Then click to position the object.

- If the scene is empty or if interactive placement is OFF, the imported object is automatically placed in the scene in the object's local coordinate space. (*Local coordinate space* is the coordinate space in which the object is modeled.) Note that if you import two objects without interactive placement and they're both modeled in the same coordinate space (for example, at the origin), they appear on top of each other.

Jump to:

- [Importing Objects \(general\)](#)
- [Importing Objects as Inlines](#)

Importing Objects

Find it: Choose *File > Import*

You can add 3D objects, 2D images, sounds, and movies to your scene, either by adding the object directly to your scene or by importing the object using an *inline* reference. If you import the object as an inline, a reference to the file containing the object is added to your scene, but the file itself isn't copied. There's no visible difference between an object imported as an inline and an object imported directly into the scene. Inlines are often used for complex geometry and for textures, to keep your file small and to reduce transmission time.

Jump to:

- [Importing 3D Objects](#)
- [Importing Images](#)
- [Importing Sounds](#)
- [Importing Objects as Inlines](#)

Importing Images

on this page: [importing an image](#) | [textures](#)

Find it: Choose *File > Import...*

(or drag and drop the file icon from the desktop into the main window)

Importing an Image

When you import an image, the image is "pasted" as a flat rectangle in the scene. This rectangle is one meter high and maintains the aspect ratio of the original image. (*Aspect ratio* is the ratio of width to height.) You can drag and drop multiple images at one time.

The image can be in JPEG, GIF, PNG, or SGI (*.rgb*, *.rgba*, or *.bw*) format.

Textures

Often, you'll import images into a Texture Editor palette and then apply this new texture to the objects in your scene.

Find it: Click the Texture Editor button on the *Looks* palette:



Drag and drop one or more image files into the palette area. If the palette is filled, you are prompted to name the new palette that is created.

Note: Textures can be expensive. Since textures can quickly bloat the size of your file, the Texture Editor issues a warning if you try to import a large texture. To turn off the warning mechanism, click the check box for *File > Warn on size* in the Texture Editor.

Jump to:

- [Importing Objects \(general\)](#)
- [Texture Editor \(quick reference\)](#)

Importing Sounds

Find it: Choose *File > Import*

(or drag and drop the file icon from the desktop into the main window)

You can import sounds into your file using the *File > Import* option, or by dragging and dropping file icons directly from the desktop.

The following directory contains the sound files included in the Cosmo Worlds distribution:

</usr/share/data/sounds/soundscheme/soundfiles/>

When you import a sound file, a red wireframe box appears in the scene until you click to add the sound. A global sound is created at the origin, pointing down the Z axis, and with spherical minimum and maximum ranges. The URL is set to the name of the file you imported, and the default name is "audioclip." A green icon indicates the location and direction of the sound. Use the *View > Show/Hide Iconic Objects* option to toggle display of sound icons.

Cosmo Worlds supports sound files in the following formats: *.wav*, *.aiff*, *.aifc*, *.au*, and MPEG audio.

After you import a sound, you'll need to create a [trigger](#) so that the sound will be played when the specified type of event occurs (clicking an object, entering a world, coming close to an object). Also, be sure to adjust the range of the sound so that it can be heard at the desired distances. See [Creating Sounds](#) for more information.

Jump to:

- [Importing Objects \(general\)](#)
- [Importing Objects as Inlines](#)

Creating a Trigger

Find it: Click the Trigger Creator button in the Sound Editor, Keyframe Animator, or Script Editor:



Sounds, animations, and scripts are all activated by some kind of trigger, which is a special node or a sensor that responds to certain types of events. The Trigger Creator button is red when the trigger needs to be created. After the trigger has been created, the button turns gray, and the handle on the trigger is depressed.

The Trigger Creator menu offers the following choices:

Collision Node

Activates when the user bumps into an object.

World Entry Script

Activates when the VRML file is loaded.

Proximity Sensor

Activates when the user approaches a certain location. (This sensor has a visible icon associated with it.)

Touch Sensor

Activates when the user clicks the specified object.

Viewpoint Binding

Activates when the browser binds to the specified viewpoint.

Visibility Sensor

Activates when a certain area in the world is visible. (This sensor has a visible icon associated with it.)

Icons

Use the *View > Show/Hide Iconic Objects* option to toggle display of the proximity and visibility sensor icons. The proximity sensor icon is a purple cube that you can scale, rotate, and translate as you would any manipulator. The sensor is activated whenever the user enters the purple cube. The visibility sensor is a red cube that you can scale, rotate, and translate. The sensor activates when the red cube is visible.

Jump to:

- [Sound Editor Quick Reference](#)

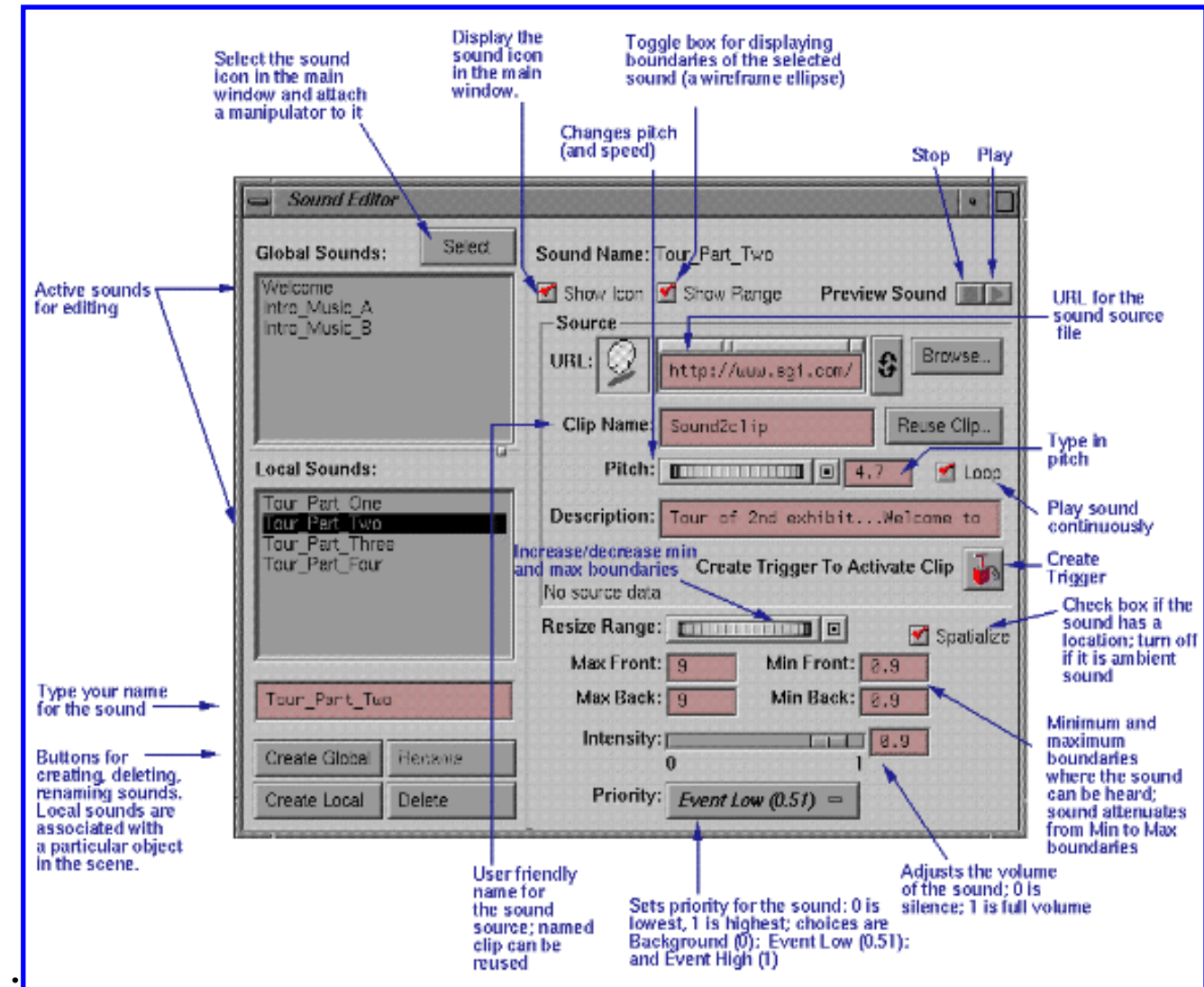
- [Keyframe Animator Quick Reference](#)
- [Script Editor Quick Reference](#)

Sound Editor



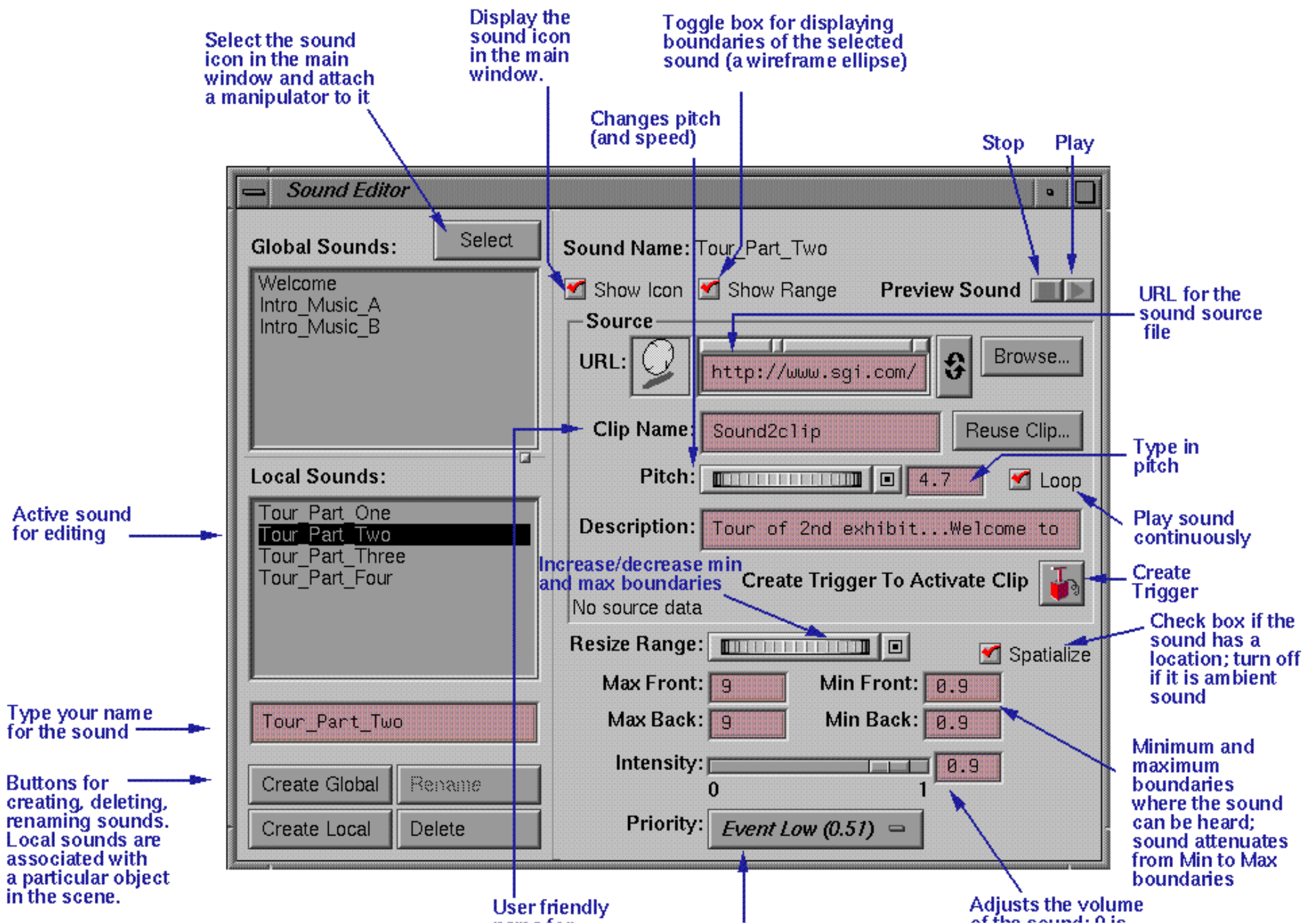
Find it: Click its button on the *Editors* palette:

Use to: Create sounds and edit their attributes.



Click image for full view.

Jump to: [Creating Sounds](#)



name for
the sound
source; named
clip can be
reused

priority
of the sound, 0 is
silence; 1 is full volume
Sets priority for the sound: 0 is
lowest, 1 is highest; choices are
Background (0); Event Low (0.51);
and Event High (1)

Creating Sounds

on this page: [boundaries](#) | [local sounds](#) | [URL](#) | [description](#) | [pitch](#) | [intensity](#) | [spatialize](#) | [stereo](#) | [recording](#) | [priority](#) | [trigger](#)



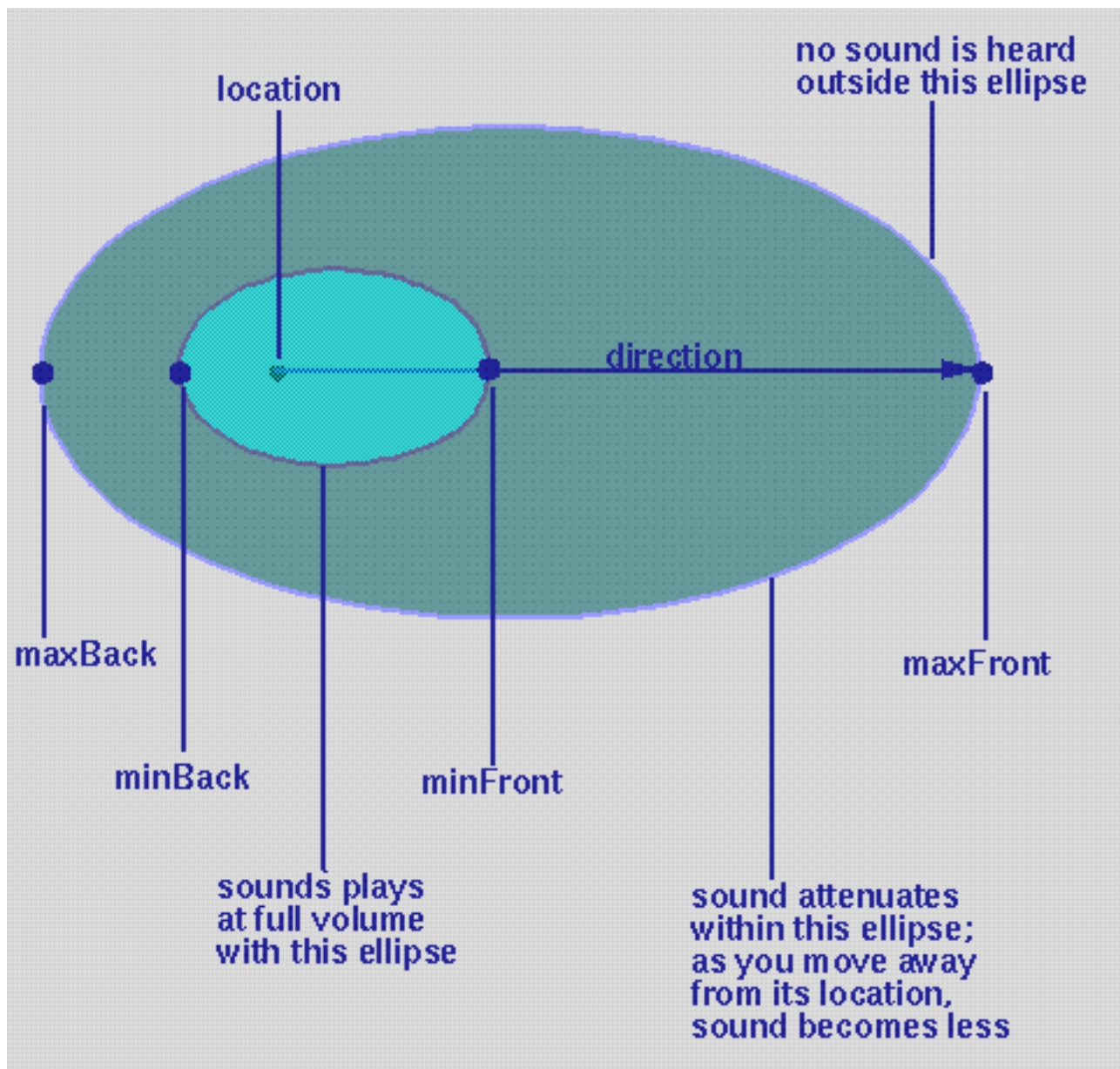
Find it: Click its button on the *Editors* palette:

Use the Sound Editor to add sounds to your scene and set attributes such as the boundaries within which the sounds can be heard, their pitch and volume, and whether they are played continuously or played only once. If you assign a name to a sound clip, you can reuse the clip multiple times in your scene. The [Trigger Creator](#) lets you specify what event triggers the sound.

You can also import sounds directly into the file (with *File > Import*) or import them as inlined files (*File > Import As Inline*). See [Importing Objects](#) for more information.

Boundaries

The Sound Editor lets you specify the boundaries of two ellipsoids, which define the area in which the sound can be heard. The inner ellipsoid is defined by *minFront* and *minBack*, as shown in this diagram:



Within the inner ellipsoid, the sound is heard at full intensity. The *maxFront* and *maxBack* values define an outer ellipsoid. Sound attenuates linearly between the inner ellipsoid and the outer ellipsoid, becoming less intense as the distance from the sound's *location* increases. The sound is not heard outside the outer ellipse.

Use the *Resize Range* thumbwheel to scale the sound range values uniformly. If you use the thumbwheel to scale the default values (where *minFront* equals *minBack* and *maxFront* equals *maxBack*), the sound is omnidirectional. In this case, the sound range is spherical rather than elliptical; this is the most efficient range for browser playback.

Creating a Local Sound

Local sounds must be placed under a grouping node in the scene hierarchy. If the *Create Local* button is grayed out, you need to select the parent group for the object or create a parent group if one doesn't currently exist.

Press the "parent" up-arrow button to select the parent group, if there is one. If the *Create Local*

button is still grayed out, choose *Edit > Add Parent Group* to create the parent group and enable the *Create Local* button.

Specifying the Sound File URL

The source file for the sound clip is specified in the *URL* field. The file can be in any format supported by the 6.2 Audio Library (*.wav*, *.aiff*, *.aifc*, *.au*). It can also be a movie format that supports sound (such as MPEG1-Systems).

Description Field

The *description* is displayed by the browser. If sounds are turned off, the browser might display this text at the time the sound would be played.

Pitch

Pitch specifies how fast the sound should be played. A value of 1 indicates normal pitch and speed. A value of 2 indicates to play the sound twice its normal speed; this results in a pitch one octave higher than normal. A value of .5 plays the sound one octave lower than normal.

Intensity

The *intensity* value adjusts the volume of the sound. 0 equals silence. 1 is full volume. For additional amplification, use the Outline Editor to specify a value greater than 1.

Spatialize

The *Spatialize* check box indicates whether the browser should position the sound from left to right. If *Spatialize* is on (the default), the browser positions the sound from left to right. Spatialized sounds should always be monaural (one channel).

If *Spatialize* is off, the sound plays at equal volume in both stereo channels. Stereo sounds will play with their normal stereo separation and balance.

To create an ambient sound, turn *Spatialize* off and then set the *Min* and *Max* ranges to encompass your entire world.

Stereo Sound

Do not use stereo sources for sounds that will be spatialized. Use stereo sources only when you are truly interested in obtaining an ambient stereo effect. Stereo sounds require twice the bandwidth of monaural sounds.

Recording Sounds

For best performance, record your sound using a sample rate of 22.05 Hz. (This is half the sample rate of a standard CD.) Use the same sample rate for all sounds in your scene. Use the MediaConvert utility to convert existing sounds to the appropriate sample rate and number of channels.

Priority

The option menu offers four choices for *Priority*:

- Background (0) - lowest priority; use for background sounds
- Event Low (.51) - sounds that play once in response to an event
- Event High (1) - sounds that play once in response to an event
- Other - use the Outline Editor to set intermediate priorities

If the browser can't play all of the currently active sounds, it plays as many sounds as it can, starting with the sounds with the highest priority.

Triggering the Sound

Use the [Create Trigger](#) button to specify which kind of event triggers the sound. Common choices are playing the sound when the user clicks an object (*Touch Sensor*); triggering the sound when the user bumps into an object (*Collision Node*); playing the sound when the user first enters the scene (*World Entry Script*); playing the sound when the user approaches a certain spot in the scene (*Proximity Sensor*) .

Jump to:

- [Global vs Local](#)
- [Sound Editor Quick Reference](#)
- [Creating a Trigger](#)

Keyframe Animator



Find it: Click its button on the *Action* palette:

on this page: [key terms](#) | [quick reference](#) | [more information](#)

Key Terms

The Keyframe Animator lets you add action to your scene: fish can swim, birds can fly, and robots can dance. You define a set of key poses at particular points in time. These poses are called *keyframes*. The Keyframe Animator then interpolates between the poses, filling in the motion to move from one pose to the next.

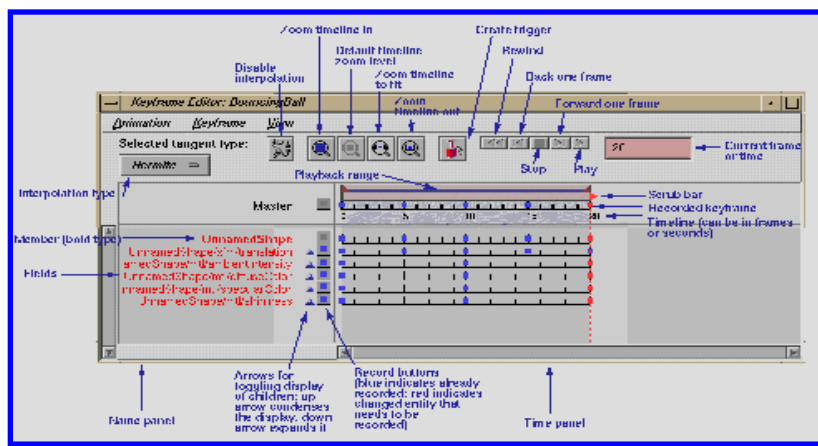
Typically, the things you'll animate are Transform properties (translations, rotations, and scales). Other properties can be animated as well, so you'll want to experiment with the effects of animating materials and colors too. In addition, you can animate viewpoints, and you can "morph" from one shape to another. The basic principle is still the same: you define the object at various keyframes, and the Keyframe Animator interpolates between the keyframes for you.

A scene can have one or more animations associated with it. Within each animation, there can be one or more *members*. Think of an animation member as one of the cast members of a play. It's an object (such as a rocking chair) or a collection of grouped objects (such as a linked chain or a human figure) that moves as a unit within the scene. Of course, the parts of a member can be articulated and can move independently of each other (when the man moves his hand, he may not move his elbow or leg). Once you've added an object as a member, all of the objects grouped underneath that object are automatically eligible to be animated.

Another key element of the animation is the *timeline*, since animation involves a series of changes over a particular time period. Each member of the animation has its own timeline.

Quick Reference Card

This diagram points out the key elements of the Keyframe Animator. Additional Help cards explain how each button works in more detail.



[Click image for full view.](#)

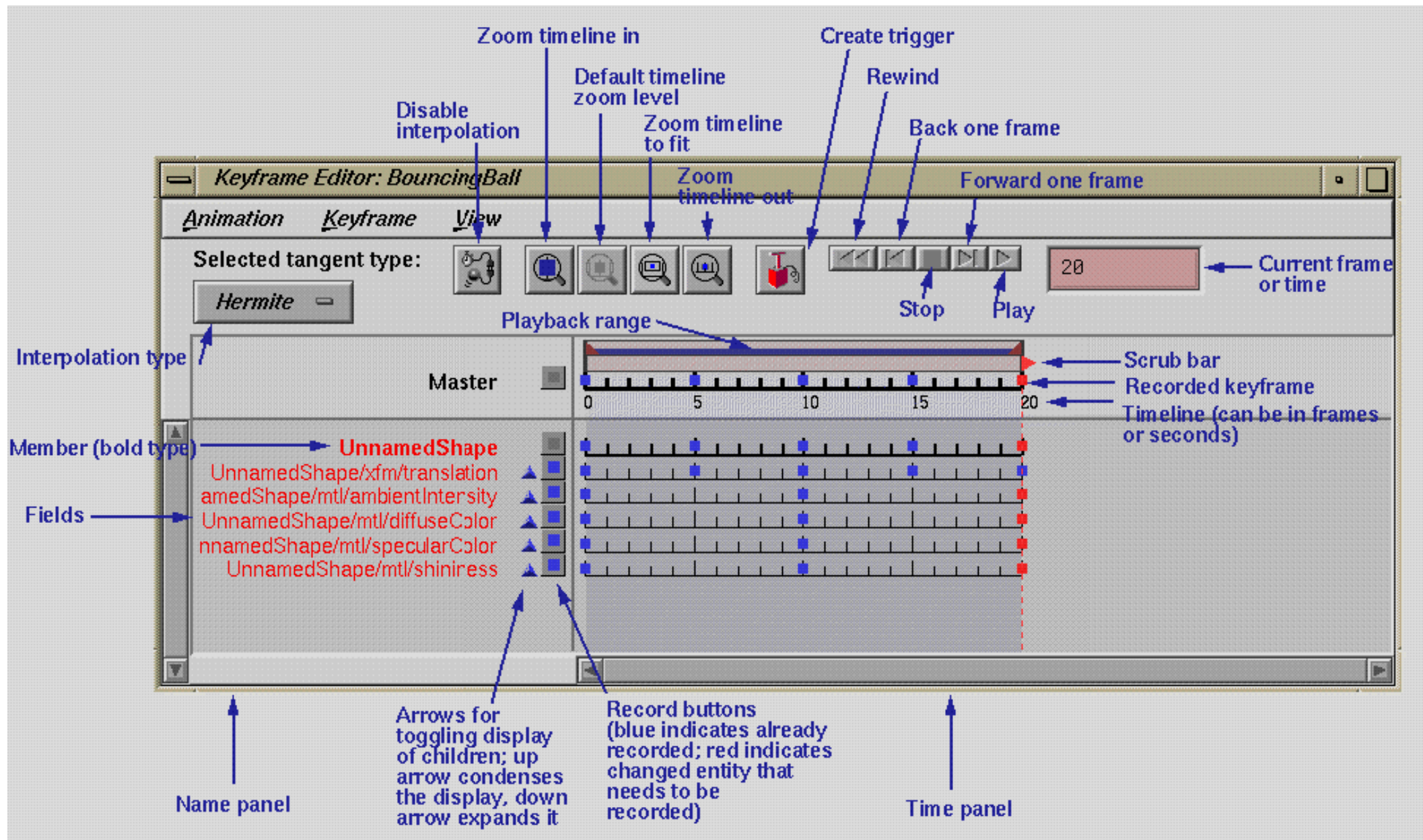
More Information

For a more detailed description of each section of the Keyframe Animator, see:

- [Animation Menu](#)
- [Keyframe Menu](#)
- [View Menu](#)
- [Names Panel](#)
- [Master Timeline](#)
- [Tangent Type](#)
- [Button Panel](#)

Jump to:

- [Keyframe Animator Tutorial](#)
- [Animating a Viewpoint](#)
- [Selecting, Copying, and Pasting Keyframes](#)



Animation Menu

Find it: Click its button on the *Action* palette:



The Keyframe Animator *Animation* menu has the following options:

Open animation (*Ctrl* + *O*) opens an existing animation.

New animation (*Ctrl* + *N*) opens a new animation. A given scene can contain a number of animations.

Delete current animation removes the animation you are currently editing. Currently, this option does not remove the trigger sensor from the scene file.

Change name changes the name of the current animation. A given scene can have more than one animation associated with it.

Add member adds a new member to an animation. An animation can contain one or more members. A member is like the cast member of a play. It could consist of one object (such as a bouncing ball), or it could consist of a number of grouped objects, such as a jack-in-the-box or a spinning top. Each member has its own timeline.

Add current viewpoint adds the current viewpoint (defined in the Viewpoint Editor) as a member of the animation. This feature allows you to [animate viewpoints](#).

Remove member removes a member from an animation. Note, however, that this option does not remove the trigger sensor for the animation.

Set duration sets the length of the animation. The maximum duration is 300 seconds.

Scale duration compresses or expands the animation to fit within a given time period. If the *Snap to frames* option is enabled, keyframes are snapped to the new grid.

Change frames per second changes the number of frames per second that are displayed on the timeline. The maximum number of frames per second is 30. Note that changing the number of frames per second does not affect the length of the animation itself (use *Set duration* to change the length). If the *Snap to frames* option is enabled, keyframes are snapped to the new grid.

Time sensor loop toggles whether the animation plays continuously when triggered or plays only once. This feature affects looping only in the browser; it does not affect looping when you press the Play button in the Keyframe Animator (the animation always loops when you press *Play*).

Jump to:

- [Keyframe Menu](#)

Animating a Viewpoint

You can animate a viewpoint using the Keyframe Animator and Viewpoint Editor. After you've defined the viewpoint in the Viewpoint Editor, add the viewpoint as a member of the animation and record different positions for the viewpoint using the Keyframe Animator. When the animation is triggered, the user's view of the scene is defined by this animation.

Animated viewpoints are useful for guided tours and for cases where you want the browser to control how the user views the scene--for example, when the user steps onto a merry-go-round or a bus. They can also be used to allow the user to travel with other objects in the scene, such as a bird or a spaceship.

Some browsers (such as Cosmo Player) interpolate between viewpoints when the user moves from one viewpoint to the next. This interpolation is strictly mathematical and may result in a path that moves the user through walls and other objects. For more explicit control over animation between viewpoints, use this feature to define the exact path the user follows through the scene.

Follow these steps to animate a viewpoint:

1. In the Viewpoint Editor, click *Create Global* to create a global viewpoint.
2. In the Viewpoint Editor, check *Lock to camera*. This step ensures that the camera and viewpoint are always synchronized.
3. In the Keyframe Animator, choose *Animation > Add current viewpoint*.
4. In the Keyframe Animator, move the scrub bar to a new time.
5. Change the view in the main window. This step edits the locked viewpoint.
6. Click the Master record button.
7. Record additional keyframes by repeating steps 4 through 6.

Follow these steps to create the trigger for the viewpoint animation:

1. In the Viewpoint Editor, toggle the *Lock to Camera* option to unlock the camera.
2. In the main window, choose *View > Show/Hide Iconic Objects / Show Viewpoints* to display the viewpoint icon.
3. In the main window, dolly out so that you can see the viewpoint icon.
4. In the main window, select the viewpoint icon.

5. Click the *Create Trigger* button. Select *viewpoint binding*. With this trigger, the viewpoint animates when the user goes to the named viewpoint in the browser.

Jump to:

- [Defining Viewpoints](#)
- [Selecting, Copying, and Pasting Keyframes](#)
- [Keyframe Animator Quick Reference](#)

Defining Viewpoints

on this page: [global](#) | [local](#) | [naming](#)

Use the Viewpoint Editor to create global or local viewpoints that will be used by a browser viewing the scene. Global viewpoints are those that you set according to your view of the scene. Local viewpoints are those that you set by grouping a camera with an object in the scene.

Also create "working" viewpoints--these are global or local viewpoints that only you can see during your construction of the scene.

To preview a viewpoint, choose *Edit > Preview* to launch a VRML browser. To move to the viewpoint within the scene, select the viewpoint and click *Jump To*.



Find it: Press the Viewpoint Editor button in the *Editors* palette.

Jump to: [Viewpoint Editor](#) for details on the buttons and menus in its window.

Global Viewpoints

Create global viewpoints to define locations from which a browser views your scene.

To create a global viewpoint:

1. Use the viewing controls or keyboard and mouse shortcuts to set up the desired view.
2. In the Viewpoint Editor, press the *Create Global* button and [name the viewpoint](#).

Local Viewpoints

Local viewpoints let you create a viewpoint associated with an object. For example, in a scene with a rocket traveling from Earth to the Moon, you can place a camera on the front of the rocket to see the view of the Moon from the rocket. Creating Local Viewpoints is important in animations, where both the camera and the object should be moved together. See [Animating Viewpoints](#) for more details.

To create a local viewpoint:

1. Select an object that you want to add a viewpoint to.
2. Open the Viewpoint Editor.

3. Press the *Create Local* button and name the viewpoint. The viewpoint appears under the same parent group as the selected object. Now when you move or rotate the object, the camera will move with it.
4. If the *Create Local* button is grayed out, choose *Edit > Add Parent Group*. This places the selected object under a parent group which includes the new viewpoint.

Once you have added a parent group above the selected object, the *Create Local* button becomes active.

To reposition the viewpoint on the object:

The usual way to reposition the viewpoint is to reposition your view using the viewer controls, and press *Update* in the Viewpoint Editor window.

When you create a local viewpoint, the viewpoint is located to reflect the current view. Because the viewpoint is grouped with an object, it will move with that object. Typically, the viewpoint is located near the object. The viewpoint icon is represented by a camera, which you can move if its current location doesn't meet your needs. You might want to move the viewpoint icon if it's too difficult to set a new viewpoint by repositioning your view of the scene using the viewer controls.

1. By default, you cannot see the viewpoint icon. Choose *View > Show/Hide Iconic Objects > Show Viewpoints*. A camera representing the viewpoint will appear in the same manipulator as the object it is grouped to.
2. Click the *Select Child* button until you see that the viewpoint icon is selected (its manipulators will appear).
3. Reposition the viewpoint icon by [dragging its manipulator](#). You can also use the [snapping tools](#) or the [Precision Placement panel](#) to accurately move the viewpoint icon.

Naming Viewpoints

When you click *Create Global* or *Create Local*, a default node name and description appear in the editor. For example, if this is the first viewpoint you have created, "VP1" and "viewpoint1" appear. VRML browsers that support viewpoints will display "viewpoint1." The node name "VP1" is used in the Outline Editor, VRML text file, and in relevant scripts and anchors. If you do not specify a description, the browser will not display the viewpoint. Not describing the viewpoint to the browser is useful for setting up "working" viewpoints for your own use while creating your scene.

To rename a viewpoint:

You can edit the node name or description by typing in a new name and pressing *Enter*.

To create a working viewpoint (one not visible to the browser):

Delete the name in the description field and press *Enter*.

Other ways to conveniently change your view during scene construction include:

- [Using the viewing buttons to set a home view and view all](#)
- [Using the construction view buttons to limit the view to a single plane](#) (Top, Right, Left, etc.)

Jump to:

- [Global vs. Local](#)
- [Animating a Viewpoint](#)
- [Viewpoint Editor](#)

Viewpoint Editor

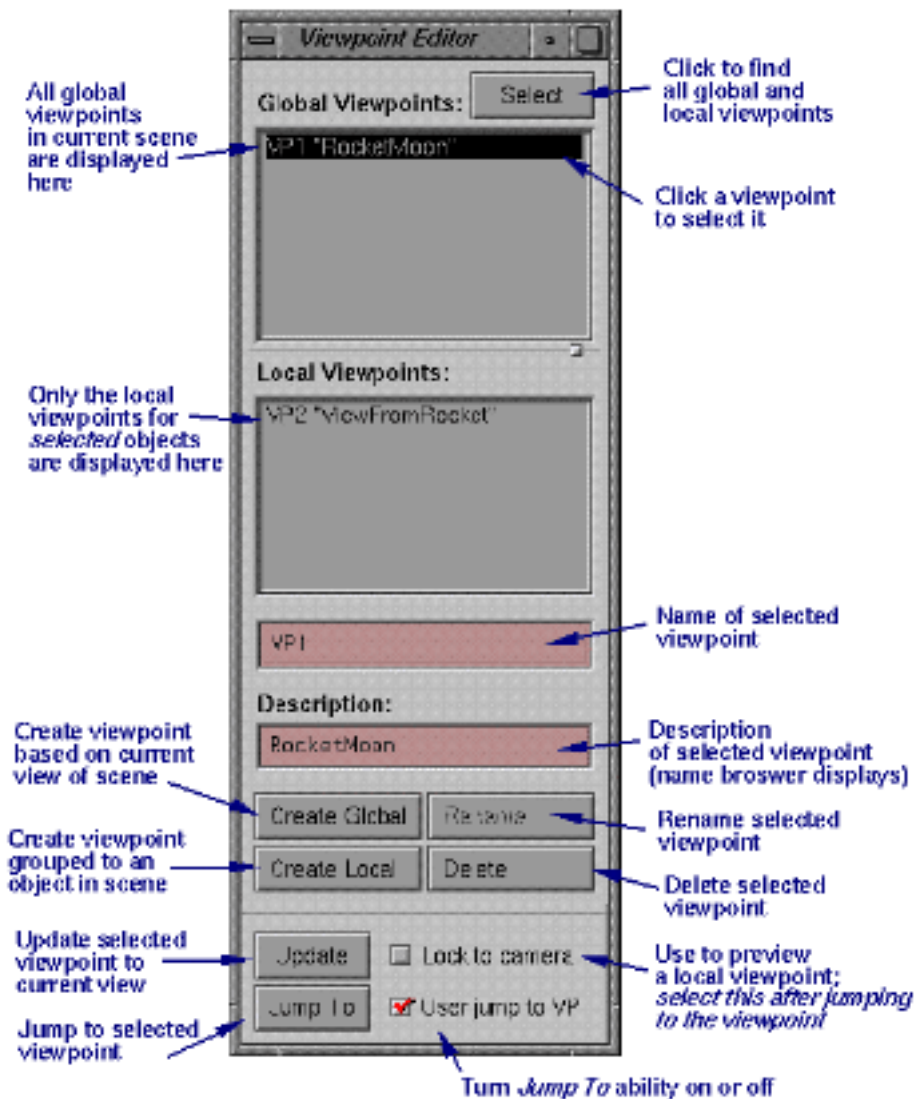


Find it: Press the Viewpoint Editor button in the Viewing tool palette.

Use to:

- [Define viewpoints for a browser](#) by creating and updating global and local viewpoints. Global viewpoints are those that you set according to your view of the scene. Local viewpoints are those that you set by grouping a viewpoint with an object in the scene.
- [Create "working" viewpoints not visible to a browser.](#)

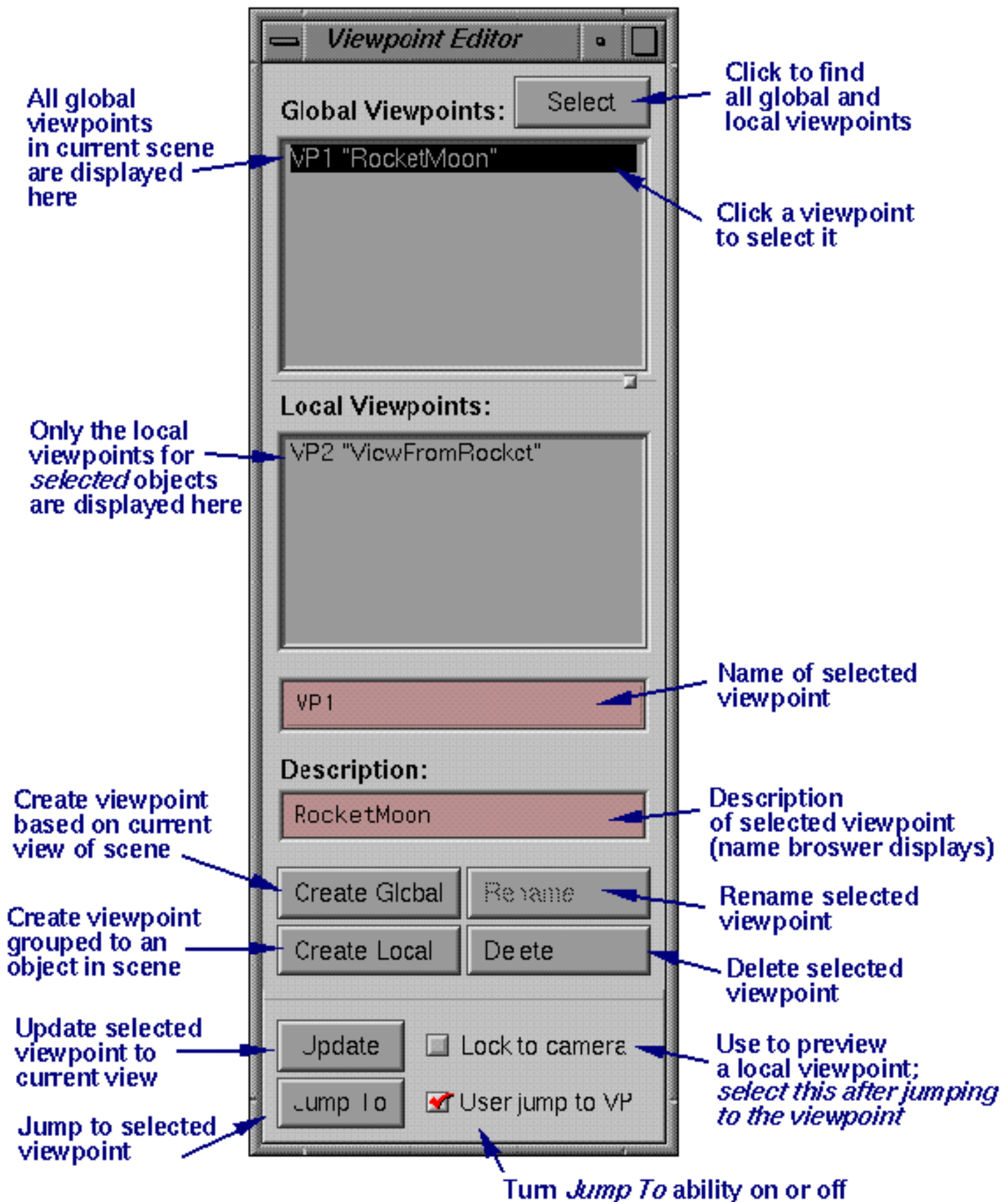
How it works:



[Click image for full view.](#)

Jump to:

- [Defining Viewpoints](#)
- [Animating a Viewpoint](#)



Selecting, Copying, and Pasting Keyframes

on this page: [selecting keyframes](#) | [defining a range](#) | [cutting](#) | [copying](#) | [pasting](#) | [try it!](#) | [copying between two animations](#)

Selecting Keyframes

To select a keyframe, click it. The selected keyframe turns red. To select multiple keyframes, shift-click additional keyframes to add them to the selection. Once you've selected keyframes, you can drag them around in time. To deselect keyframes, click on an empty part of a timeline

You can select at any level of line (master, member, object, property, field). Selecting on a higher-level line is the same as selecting all keys at the same time on lower-level lines. Conversely, when you select on a lower-level line, that keyframe and all of its parents become highlighted.

Defining and Using a Range

The **range slider** is a blue line that rests inside a pink trough. When you move the cursor into this trough, it highlights. The default range slider runs the full length of the animation, and you can use the *View > Reset range* option to reset the range to the length of the full animation. Clicking to the left of the range slider extends the blue bar to the left. Clicking to the right of the range slider extends the blue bar to the right. A light gray shading in the time panel indicates the selected range. You can also drag either end of the bar to the correct keyframe. Dragging on the middle of the bar moves the bar to a different part of the animation without changing its length.

Use the range slider to define a range of keyframes you want to cut or paste. Then use the *Keyframe > Select in range* option to select the keyframes in that range.

Tip: Another useful option that uses the range slider is *View > Limit playback to range*. Specify a range with the range slider and enable this option. When you play a long animation, this option enables you to view only the parts you're interested in.

Cutting

Choose *Keyframe > Cut* (or press *Ctrl-x* or *Delete*) to cut selected keyframes.

Copying

Choose *Keyframe > Copy* (or press *Ctrl-c*) to copy selected keyframes.

Pasting

The Keyframe Animator has three paste options. All three options in the *Keyframe* menu use the keyframes currently in the paste buffer.

Paste overlay at current time (Ctrl-v)

pastes the selected keys at the current time. If the keys land at the same time as existing keys, the pasted keys replace the existing keys. If there is no new key to replace an existing key, the existing key is preserved (that is, you'll have a mixture of old and new keys).

Paste replace range (Shift-Ctrl-v)

deletes everything currently in the range and pastes in the new keys. This option is used to replace a section of animation with a new section. If the Paste selection doesn't fit into the specified range, it is truncated.

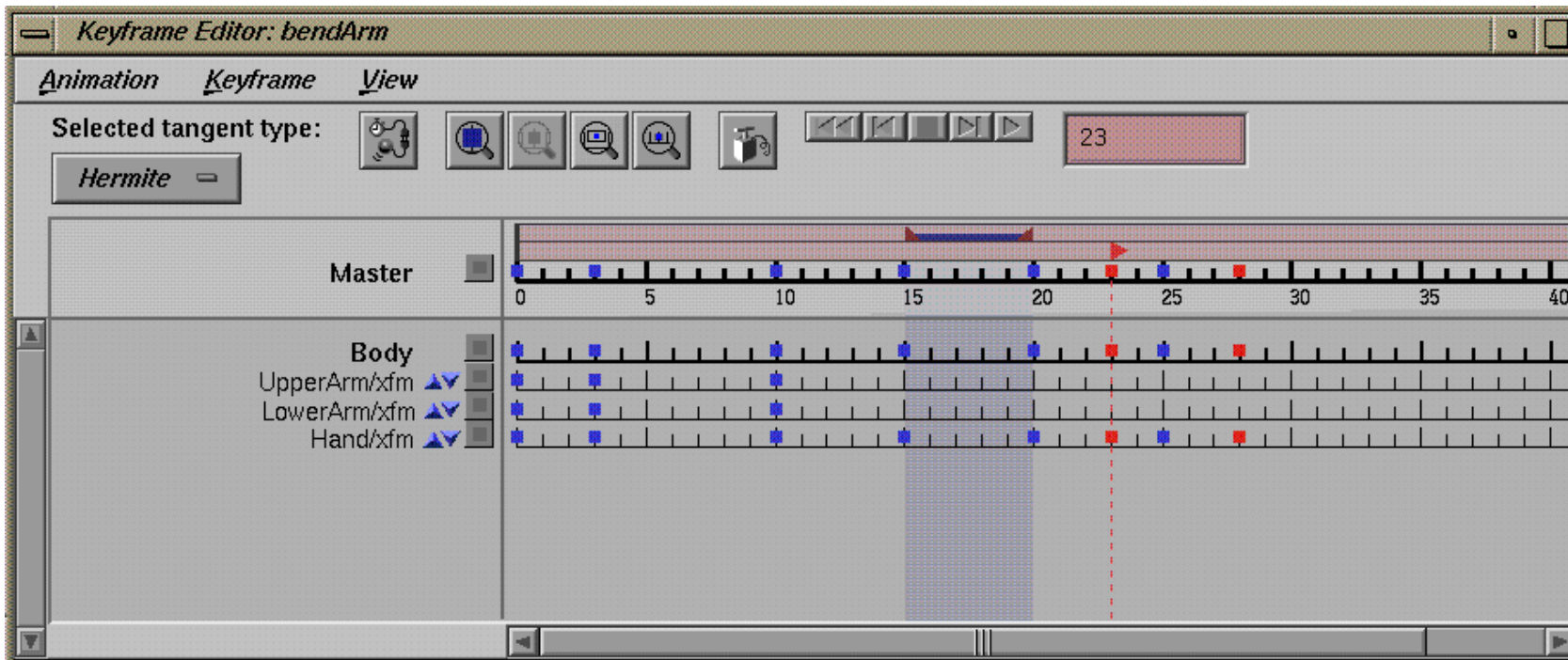
Paste insert at current time

inserts the Paste selection at the current time, extending the animation the length of the pasted section.

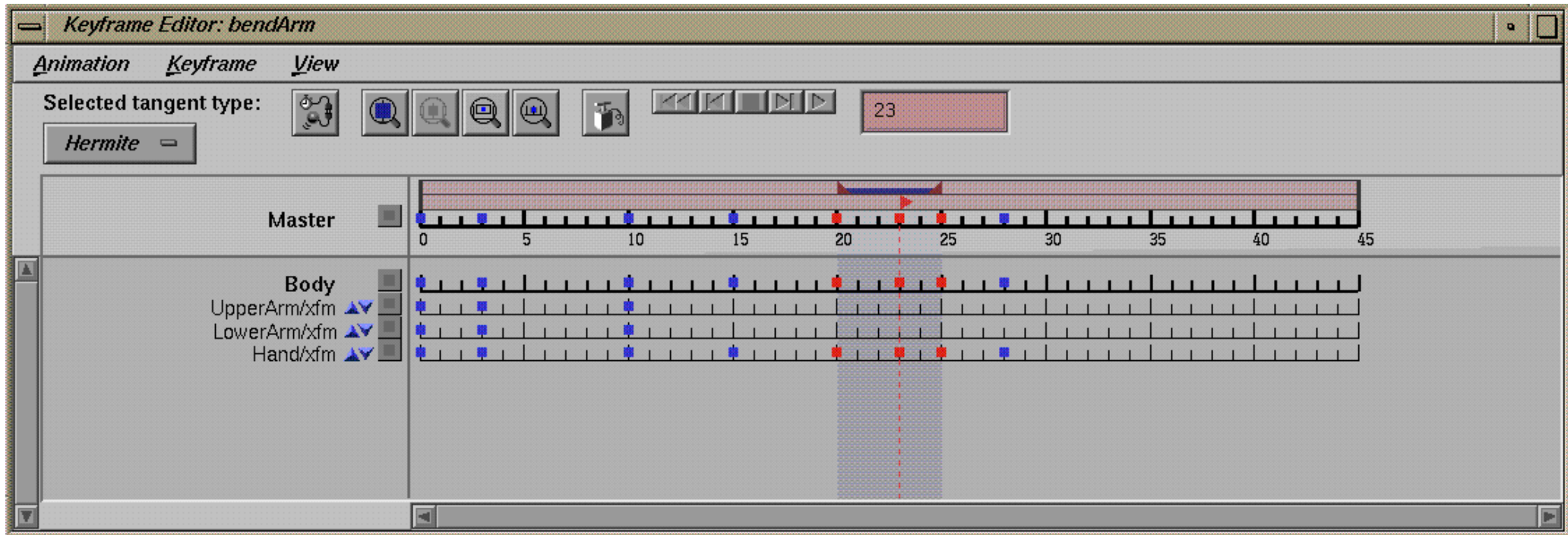
Try It!

This tutorial illustrates use of the three paste options.

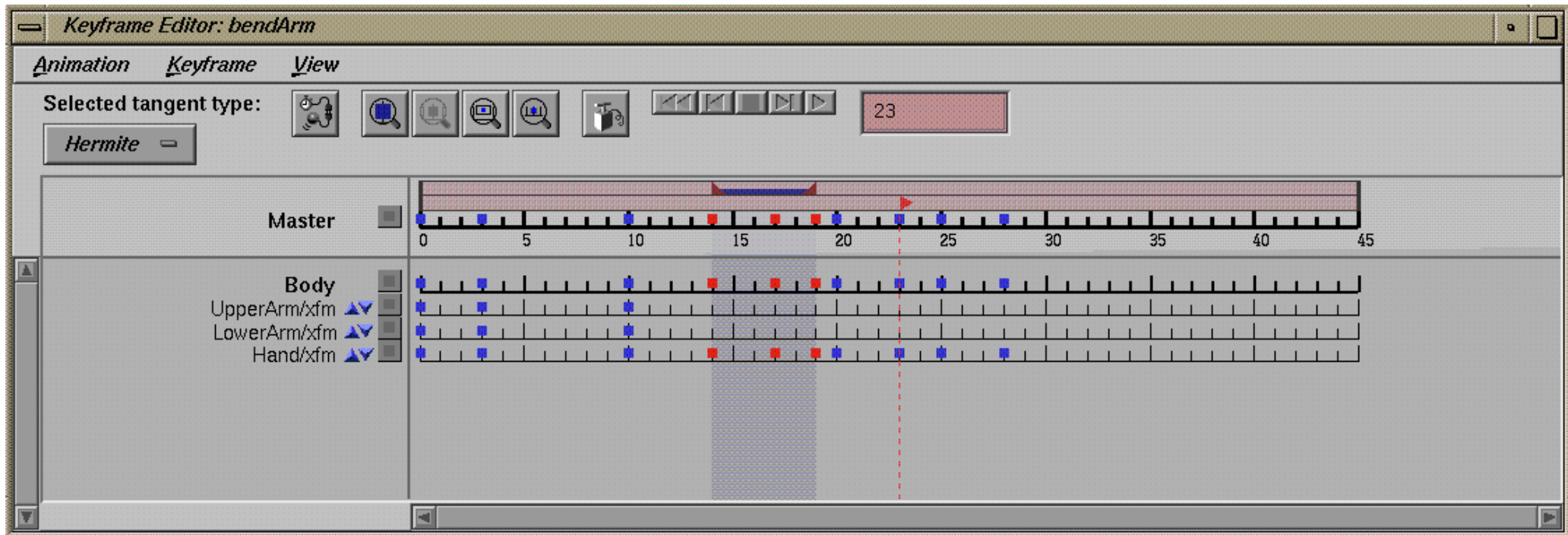
1. Open the file `/usr/share/Insight/library/SGI_bookshelves/Help/books/CosmoWorlds_UG/Models/anim.wrl`.
2. Start the Keyframe Animator by pressing the bouncing ball icon.
3. Choose *Animation > Open Animation*. Double-click the *bendArm* animation to open it.
4. Choose *Animation > Set duration* and specify 45 frames.
5. Move the range slider to extend from frame 15 to frame 20. Then choose *Keyframe > Select in range*. The keyframes from frames 15 to 20 turn red to show they're selected.
6. Press *Ctrl-c* to copy the selection.
7. Move the current time marker (the red triangle) to 23.
8. Press *Ctrl-v* to paste the overlay at the current time. The new keyframes are pasted at the current time. If new keyframes fall on existing keyframes, the new ones replace the old. In this example, the old keyframe at frame 25 is retained because there was no new value at that time to replace it.



9. Select a new range of keys to copy: frames 20 to 25. Press *Ctrl-c* to copy the selection.

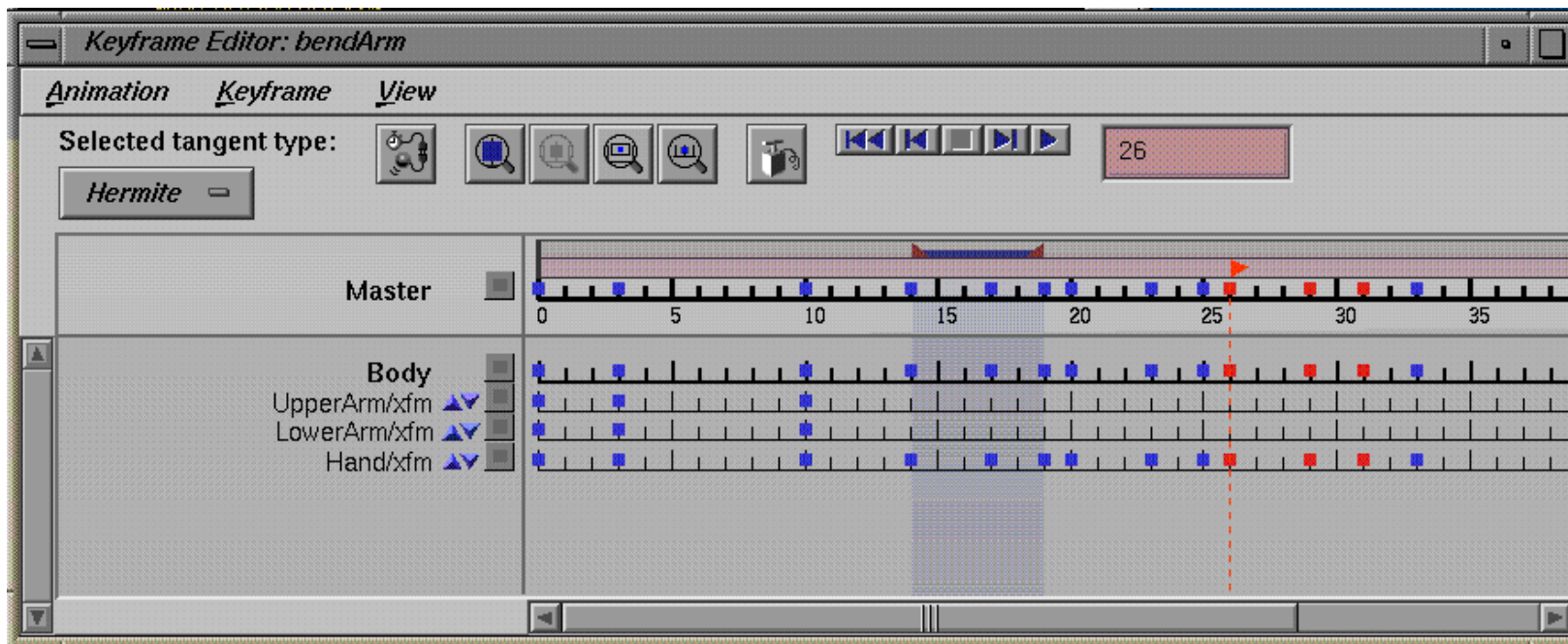


10. Move the range slider to extend between frames 14 and 19.
11. Choose *Keyframe > Paste replace range*. The values in the pasted selection completely replace the values in the selected range. In this case, the keyframe at frame 15 was removed because it was in the paste range.



12. Move the current time marker to frame 26.

13. Choose *Keyframe > Paste insert at current time*. The selected keyframes (frames 20 to 25, from step 9) are inserted at the current time, and the animation is extended to accommodate the new frames.



Copying between Two Different Animations

Note that you can also copy from the same object in one animation to another animation. Select and copy the keyframes in the first animation. Then open the second animation and paste the selected keyframes. For example, if you have a sphere in one animation, you can copy to a sphere in a second animation. If the member in the second animation does not already have an animation associated with it, the lines will be created for it when you paste the animation.

This feature enables you to copy poses from one animation to another--perhaps to duplicate starting and ending poses.

Jump to: [Keyframe Animator Quick Reference](#)

Manipulating Objects

There are several ways to manipulate objects in Cosmo Worlds. You'll always follow these two basic steps:

1. Select an object or part of an object to move or edit.
2. Translate, rotate, or scale the object interactively or by snapping objects to each other or to a grid. You can also specify a transform more precisely by entering values in the Precision Placement panel.

Jump to:

- [Tools to Use: Manipulating Objects](#)
- [Selecting and Grouping Objects](#)
- [Moving, Resizing, and Rotating](#)
- [Snapping to Align Objects](#)
- [Layout Tools](#)
- [Precision Placement Panel](#)

Tools to Use: Manipulating Objects

on this page: [starting out](#) | [simple transforms](#) | [snapping and alignment](#)

Starting Out

Check view

Before manipulating a model, [check the view that you are in](#). Construction views constrain movement and edits to a single plane.

[Select all or part of an object](#)

Click an object to select it or use *Select Parent/Select Child* buttons on menu bar.



Parents and children are parts of [group hierarchies](#). You can also use the Outline Editor to select within a group hierarchy.

[Group Objects](#)

Shift-click to select multiple objects. Choose [Edit > Group](#) to group objects. Use [Edit > Add to Group](#) to add a new object to an existing group.

Simple Transforms

[Resize an object](#)

Interactively resize by dragging white cubes for a uniform scale. *Shift*-drag to stretch in a single direction. *Ctrl*-drag to uniformly scale about the opposite corner. *Ctrl-Shift* to stretch about opposite corner.

Specify X,Y, Z coordinates--use the [Precision Placement panel](#). Choose *Layout > Precision Placement*.

[Move an object](#)

Interactively move by selecting and dragging transform box face.

Nudge--use arrow keys or choose *Layout > Nudge > Left a Little, Right a Little, etc.*

Specify X,Y, Z coordinates--use the [Precision Placement panel](#). Choose *Layout > Precision Placement*.

[Rotate an object](#)

Interactively rotate by selecting object and dragging green knobs.

Specify X,Y, Z coordinates--use Precision Placement panel. Choose *Layout > Precision Placement*.

Snapping and Alignment

Snap two objects together

Use *Layout > Activate Snap Target* or press its button from the Layout palette:



Drag target to object. Select object to move, press **Ctrl-m to move selection to target**, or choose *Layout > Move Selection to Snap Target*.



Also use **Ctrl-click middle to activate the Source snap target**.

Layout > Activate Scale Snapping --**scale desired features** on a selected object (for example, a cylinder's height) to the same measurement as that of an object on which the Snap Target is placed.

that of another object's edge; place the Snap Target on the non-selected object's edge. to match size of target object. Good for making even walls or boxes.

Layout > Edit Snapping Options--**set tolerances** for how close to the snap target the manipulator must be before snapping occurs. For example, adjust the rotation tolerance to a larger number to make snapping occur when the rotation is further from a 90 degree rotation increment.

Align to Grid

Use *Grid*  or *Angle* .

Jump to:

- [Manipulating Models](#)
- [Moving, Resizing, and Rotating](#)
- [Selecting and Grouping Objects](#)
- [Snapping to Align Objects](#)
- [Layout Tools](#)
- [Precision Placement Panel](#)

Tools to Use: Viewing

on this page: [the hand icon](#) | [choose a viewer](#) | [free or construction views](#) | [viewing buttons](#) | [finding the palette and toolbar](#)

View with the Hand Icon

Choose the hand icon found on the viewing toolbar to move your view of the scene:



Choose the arrow icon to manipulate and edit objects:



Hold down *Alt* to temporarily switch from the arrow icon to the hand icon. *Esc* toggles between the two modes.

Choose a Viewer

Use the [Examiner Viewer](#) to rotate a scene. Use the [Walk Viewer](#) to navigate through a scene or pan side to side. Choose *View > Examiner Viewer*, or *View > Walk Viewer*. Access a [menu](#) from either viewer by holding the right mouse button over the viewing area.

Free or Construction Views

Construction views use the [Plane Viewer](#) with predefined, orthographic (2D) views that limit object manipulation to a single plane. For example, Top View lets you move objects only parallel to the top plane of the scene. Choose *View > Set Working View > Front, Back, Left, etc.* Or choose an icon from the Viewing palette:



Use construction views for precise alignment and inspection of objects.

A **free view** is an unconstrained perspective projection, or 3D view. From a construction view, return to the last perspective view used by pressing the Free View button found on the Viewing palette:



You can change from a perspective view to an orthogonal view without the limitations of the plane viewer by pressing this button found on the Viewing toolbar:



Other Viewing Buttons

View All moves the camera to include all objects in scene (use in Free or Construction views).



Set Home View sets the viewer's new home position.



View Home returns you to your home view.



Seek to View brings the camera closer to an object you click on. Click again before the movement stops to seek toward a different object or point. You can set the amount of time it takes to seek to an object and the distance travelled by using the [Preferences panel](#). Choose *Viewer Menu > Preferences*.



Finding the Viewing Palette and Toolbar

To open the Viewing Palette, choose *Palettes > Viewing*:



The Viewing toolbar is found on the right side of the main window:



Jump to:

- [Viewing Objects and Scenes](#)
- [Defining Viewpoints](#)
- [Keys & Shortcuts for Viewing](#)

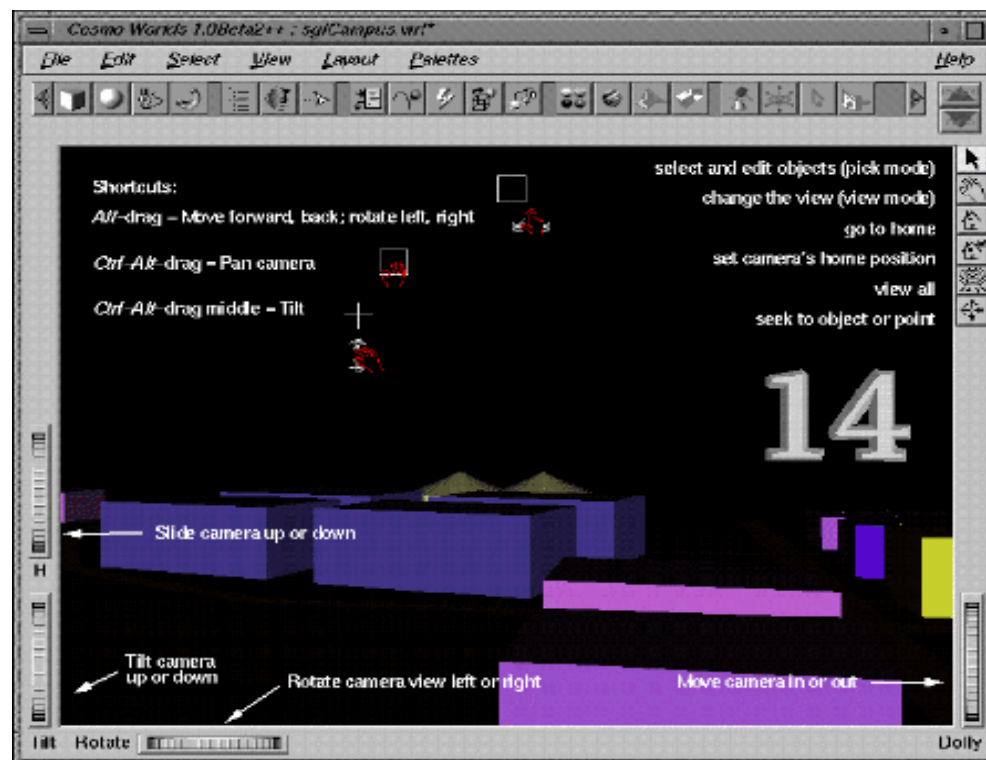
Walk Viewer

on this page: [how it works](#) | [viewer controls and menus](#) | [shortcuts](#)

Use to: Navigate, dolly, and pan when viewing a scene. For example, use the Walk Viewer to look at architectural models from the inside, or to walk between buildings or across landscapes.

Find it: Choose *View > Walk Viewer*, or use *Ctrl-k*

How It Works




[Click image for full view.](#)

The Walk Viewer uses a walking or driving metaphor to move the camera through the scene.


The Walk Viewer is one of two viewers in Cosmo Worlds. The other viewer, called the [Examiner Viewer](#), lets you examine objects by rotating, dollying, and panning.

Both the Examiner Viewer and the Walk Viewer have a menu that you access by holding your right mouse button over the viewing area. Use this [Viewer Menu](#) to [change the draw style](#) (rendering characteristics) and [preferences](#).

Trick: Click the *Seek* button , then click an object or point in your scene to move quickly toward

it. Click again before the movement stops to seek toward a different object or point.

Viewer Controls and Menus

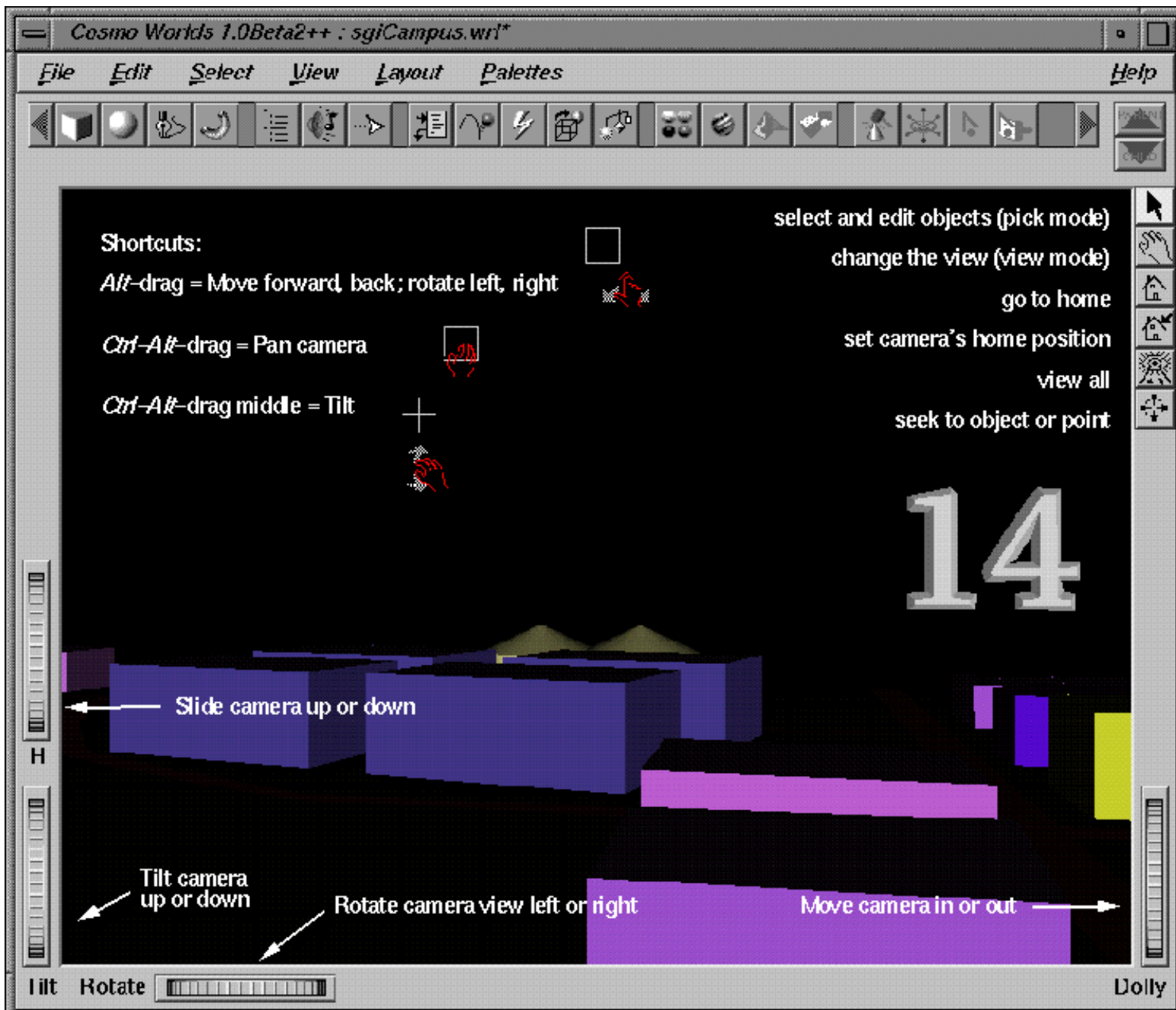
If you haven't looked at the [Walk Viewer diagram](#), do so now. This image briefly describes the Walk Viewer's controls and toolbar buttons. For a full description of the viewer toolbar and the View palette, see [Tools to Use: Viewing](#). There is also a viewer menu (accessed with the right mouse button). Use this to change the draw style and viewer-related preferences. See [Viewer Menu](#) for details.

Shortcuts

You can use keyboard shortcuts instead of the viewer controls. See [Keys & Shortcuts for Viewing](#).

Jump to:

- [Viewing Objects and Scenes](#)
- [Tools to Use: Viewing](#)
- [Keys & Shortcuts](#)







Viewer Popup Menu

Use to: Set options and select functions for viewers.

Find it: When in the [Examiner Viewer](#), [Walk Viewer](#), or [Plane Viewer](#) hold down the right mouse button over the work area in Cosmo Worlds. The menus for all viewers are similar.

How it works:



- **Functions** lets you:
 - Move to your home position. Also access on the Viewing toolbar by pressing:
 
 - Set a new home position. Also access on the Viewing toolbar by pressing:
 
 - View all the objects in your scene. Also access on the Viewing toolbar by pressing:
 
 - Seek to an object or point by choosing this menu item and then selecting the point. Also access on the Viewing toolbar by pressing:
 
 - Copy a current camera position which can later be pasted in another viewer by using *Paste Viewpoint*.
- **Draw Style** lets you change the way Cosmo Worlds renders objects. See [Changing the Draw Style](#) for details.

- The **Viewing** command is equivalent to the [pick and view buttons](#) on the toolbar. When a checkmark appears next to Viewing, you are in viewing mode. The cursor resembles a hand and you can change your view of the scene. When the checkmark is absent, you are able to select and edit models.

Shortcut: In selection mode, use *Alt* to temporarily enter View mode.

- **Decoration** lets you hide or display all of the controls that appear around the viewing area.

Turn on decoration (checkmark appears) to see the toolbar, zoom, rotation, and navigation controls in the viewer windows for editing tools. Turn off decoration (no checkmark) to hide the viewer controls around the viewer window.

- The **Headlight** command lets you turn on and off the headlight, which is a single directional light positioned from the camera at the scene. Turn the headlight off to see only those lights you have added to the scene. (Use the [Light Editor](#) to add lights to your scene.)

When you open a scene without lights, Cosmo Worlds automatically sets the headlight to *on*. When you open a scene with lights, Cosmo Worlds automatically sets the headlight to *off*. Note that VRML browsers have their own way to use headlights. Turning the headlight on or off in Cosmo Worlds does not set the headlight for a browser. To set the headlight for a browser, see [Adding Navigation Information](#).

Note: The headlight is reset to *on* each time you open a file.

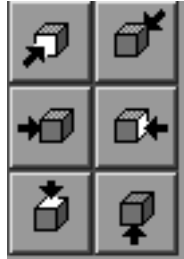
- **Preferences** has a pop-up window that lets you set several options like seek animation time, camera zoom angle, and auto clipping planes. See [Viewer Menu Preferences](#) for details.

Plane Viewer

on this page: [how it works](#) | [viewer controls and menus](#) | [shortcuts](#)

Use to: Reposition the camera when using [construction views](#) for precise alignment and inspection of objects. Some editors, like the [Extrusion Editor](#), also use the plane viewer.

Find it: Choose *View > Set Working View* , or click one of the construction view buttons:



How It Works

The Plane Viewer limits the camera to panning, zooming, and rotating within a specified plane (top or side, for example). The view changes from that of a perspective projection (3D) in Free View, to that of an orthographic projection (2D) in a construction view. Return to the last 3D view by pressing the Free View button:



Viewer Controls and Menus

The thumbwheel controls change in the Plane Viewer to *transX* (translate the camera along the X axis), *transY* (translate the camera along the Y axis), and *Zoom*. You can also rotate within the plane by pressing *Ctrl-Alt*-drag middle.

Shortcuts:

- *Alt*-drag to zoom
- *Ctrl-Alt*-drag to pan (slide the camera)
- *Ctrl-Alt*-drag middle to rotate within the plane (cursor changes to an anchor)

For a full description of the viewer toolbar and the View palette, see [Tools to Use: Viewing](#). There is also a viewer menu (accessed with the right mouse button). Use this to change the draw style and viewer-related preferences. See [Viewer Menu](#) for details.

Shortcuts

You can use keyboard shortcuts instead of the viewer controls. See [Keys & Shortcuts for Viewing](#).

Jump to:

- [Viewing Objects and Scenes](#)
- [Tools to Use: Viewing](#)
- [Keys & Shortcuts](#)

Extrusion Editor

on this page: [how it works](#) | [the window](#) | [example](#)



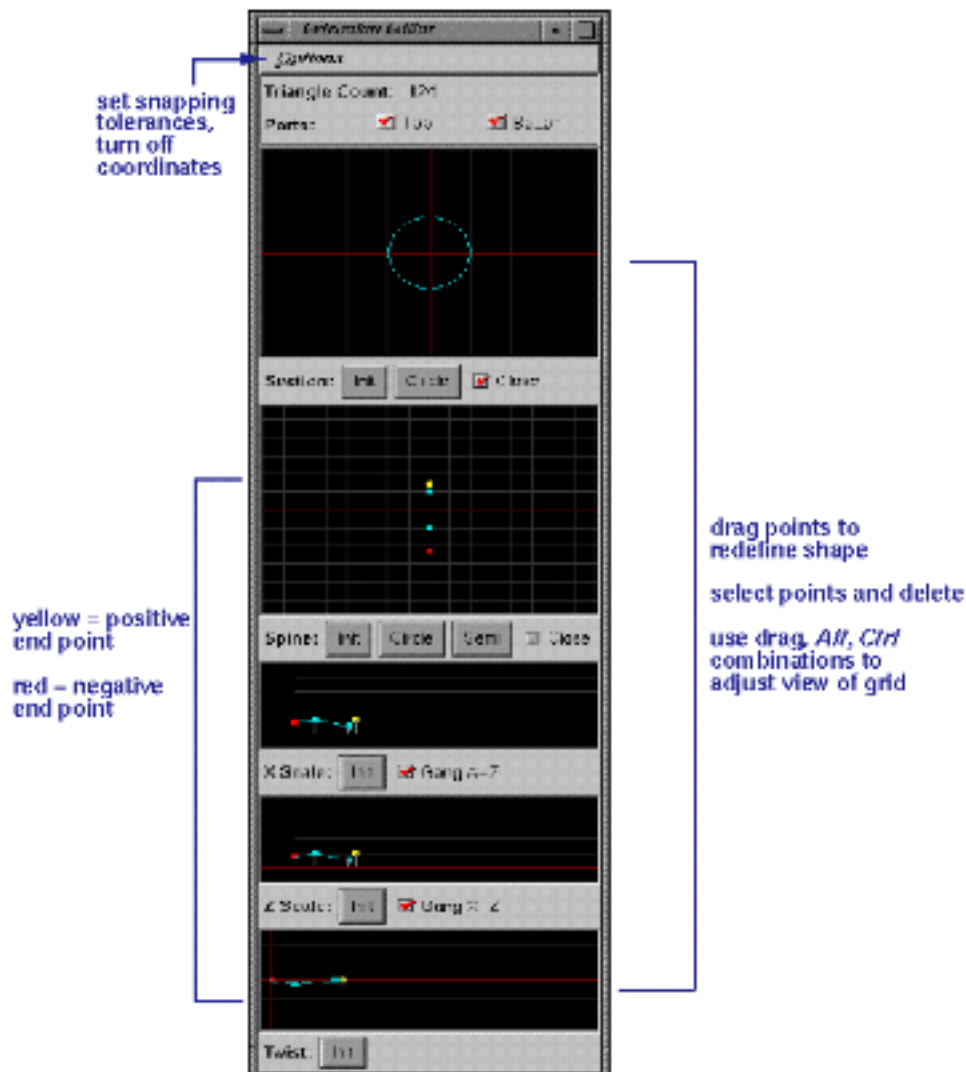
Find it: Click its button on the *Create* palette:

- This creates a default shape which you can place in your scene. After placing, the Extrusion Editor appears.
- You can also select a shape created with the Extrusion Editor and press *Ctrl-e* to launch the Extrusion Editor for further editing.

Use to:

- Create tube-shaped extrusions to form objects like vases, fountains, and columns.
- Create triangular or cylindrical tube-shapes.
- Create torus shapes.
- Create shapes with more complex spines by defining connecting points on the grid.

How It Works



[Click image for full view.](#)

- The Extrusion Editor contains five grids: *Section*, *Spine*, *XScale*, *ZScale*, and *Twist*.
- Change your view of the Spine grid by using [Examiner Viewer shortcuts](#), or select *Decoration* in the [Viewer pop-up menu](#).
- Change your view of the *Section*, *XScale*, *ZScale*, and *Twist* grids by using [Plane Viewer shortcuts](#), or select *Decoration* in the [Viewer pop-up menu](#).
- The grids show coordinate numbers by default when you pass the cursor over a grid. Use the *Options* menu to turn off the coordinates.
- By default, dragging snaps endpoints to the grid.
- Undo and redo operations by using *Ctrl-z* and *Shift-Ctrl-z*.
- The triangle count of the selected shape appears at the top of the Extrusion Editor window.
- **Note:** You cannot edit an extrusion further in the Extrusion Editor after it has been modified

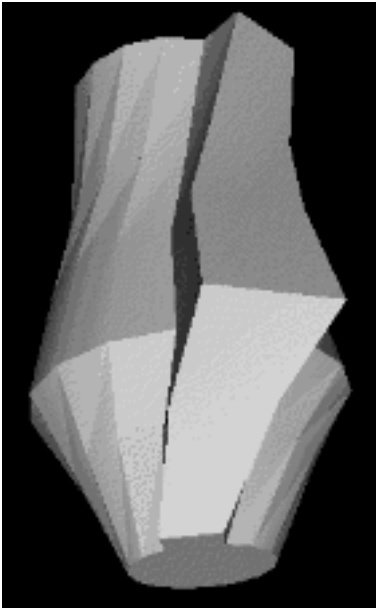
in the PEP editor.

The Window

- **Options:** Set snapping tolerances and turn off coordinates.
- **Parts:** Deselect *Top* or *Bottom* to create a shape with exposed edges. For example, a vase has one open end with exposed edges. (The top endpoint appears yellow; the bottom is red.)
- **Section:**
 - Drag points, click to create new points, or delete points to change a shape's appearance.
 - Press *Init* to clear the current shape.
 - Press *Circle* to change the shape from the default triangle to a circle (this increases the triangle count).
 - Select *Close* to generate a new segment between the endpoints of the shape.
- **Spine:**
 - Click in the Spine grid to create new segments on the shape. Drag points to resize the segments. Delete a point by clicking on it and pressing *Backspace*.
 - Drag the segment endpoints to resize the segments.
 - A shape with no spine appears flat.
 - Press *Init* to clear the segments and endpoints from the grid.
 - Press *Circle* to create a torus shape.
 - Press *Semi* to create a half-torus shape.
 - Select *Close* to generate a new segment between the endpoints of the shape.
- **XScale:**
 - Drag points to change the x-scale of the cross section about the spine point for that section.
 - Press *Init* to clear all points.
 - Deselect *Gang X-Z* to move segments independent from ZScale.
- **ZScale:**
 - Drag points to change the z-scale of the cross section about the spine point for that section.
 - Press *Init* to clear all points.
 - Deselect *Gang X-Z* to move segments independent from XScale.
- **Twist:**
 - Drag a point to twist its corresponding segment.
 - Press *Init* to clear all points.

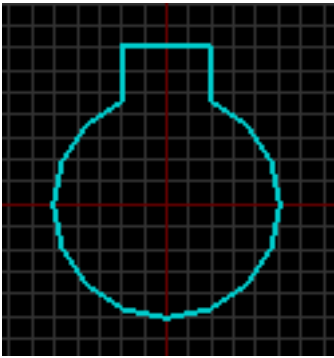
Example

The following shape was created in the Extrusion Editor to illustrate the relationship between the shapes in the editor windows and the extrusion they create:

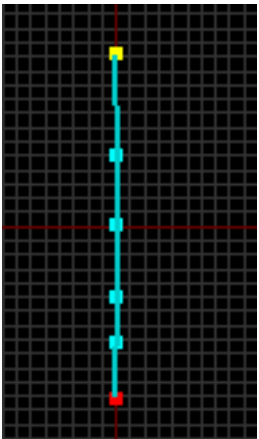


Example Extrusion

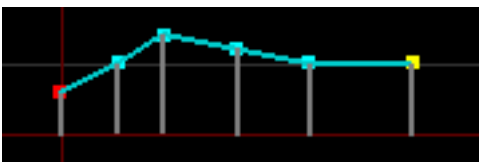
Here are the shapes that create the example extrusion:



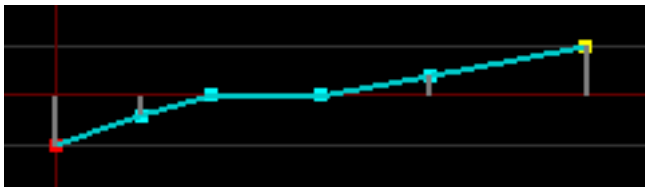
Section



Spine



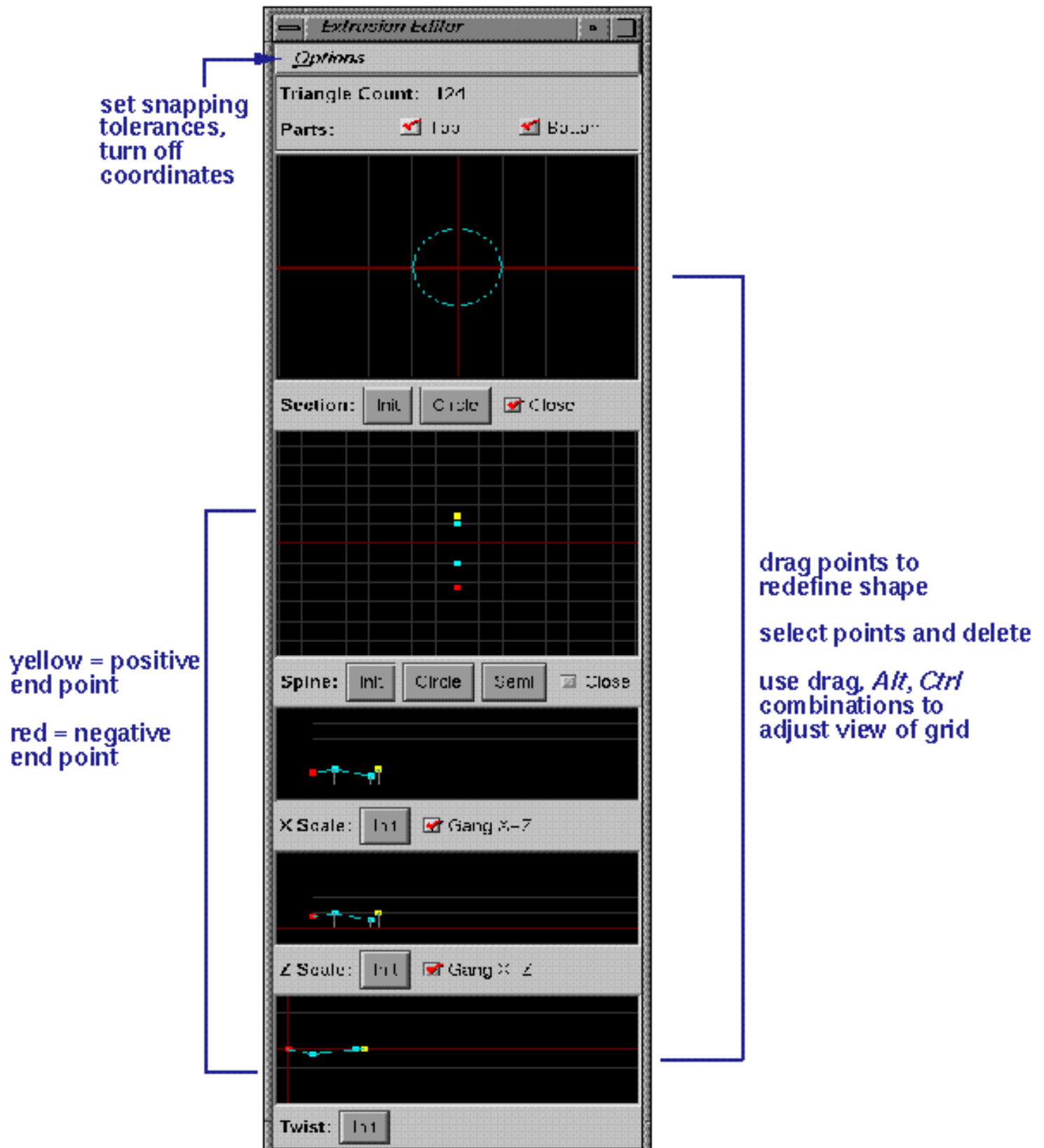
X and Z scale (ganged, so they're both the same)

*Twist*

Finally, here's what the inside of the extrusion looks like. The darker shapes were manually inserted to show you exactly how the twist moves the spine. Compare this to the solid image of the extrusion above.

*Inside of Example Extrusion*

Jump to: [Extruding Points, Edges, and Polygons](#)



Keys & Shortcuts

on this page: [open, import, save](#) | [view](#) | [place](#) | [select & group](#) | [move](#) | [layout](#) | [resize](#) | [rotate](#) | [edit](#) | [pep scale & rotate](#) | [animate](#) | [scripts & routes](#) | [abbreviations](#) | [hints](#)

Open, Import, Save

- New = **Ctrl-n**
- [Open](#) = **Ctrl-o**
- [Import](#) = **Ctrl-i**
- [Save](#) = **Ctrl-s**
- [Preview](#) = **Ctrl-r**
- Exit = **Ctrl-q**

[View Controls](#)

- Walk Viewer = **Ctrl-k**
- Examiner Viewer = **Shift-Ctrl-e**
- Temporarily switch from selection mode to view mode = **Alt**
- Toggle between selection mode and view mode = **Esc**
- Free view = **F2**
- View front = **F3**
- View back = **F4**
- View left = **F5**
- View right = **F6**
- View top = **F7**
- View bottom = **F8**

[Examiner Viewer](#)

- Rotate camera around objects in scene = **Alt-drag** or **Alt-arrow keys**
- Pan to slide camera left, right, up, down = **Ctrl-Alt-drag** or **Ctrl-Alt-arrow keys**
- Dolly = **Ctrl-Alt-drag middle**

[Walk Viewer](#)

- Move forward or back, turn left or right = **Alt-drag**
- Pan to slide camera left, right, up, down = **Ctrl-Alt-drag** or **Ctrl-Alt-arrow keys**
- Tilt up or down = **Ctrl-Alt-drag middle**

[Plane Viewer](#)

- Dolly = **Alt-drag**
- Pan = **Ctrl-Alt-drag**
- Rotate in same plane = **Ctrl-Alt-drag middle**

Place on Import, Paste, or Create

- Move and resize = **drag before releasing to place**
- Place without moving or resizing = **click**

Select & Group

- Select all = **Ctrl-/**
- Deselect all = **Ctrl-**
- Select multiple = **Ctrl-click** or **Shift-click**
- Group = **Ctrl-g**
- Ungroup = **Shift-Ctrl-g**

Move Selection (drag faces of the manipulator)

- Move in plane of surface = **drag**
- Constrain move in plane of surface = **Shift-drag**
- Move perpendicular to surface = **Ctrl-drag**
- Nudge in plane of front-facing surface = **arrow keys**
- Nudge perpendicular to front-facing surfaces = **Ctrl-up arrow** or **Ctrl-down arrow**

Layout

- Place yellow Snap Target = **click middle on object; also drag to place, Shift-drag makes target "stick" to features**
- Put away Snap Target = **click middle over background**
- Constrain Snap Target to vertices, edges, centers = **Shift-click middle on object**
- Place green Snap Source = **Ctrl-click middle; also drag to place, Shift-drag makes source "stick" to features**
- Put away Snap Source = **Ctrl-click middle**
- Move selection to snap target = **Ctrl-m**
- Move selection center to snap target = **Shift-Ctrl-m**
- Move grid by increments = **drag**
- Move grid freely = **Ctrl-drag**

Resize (drag corner boxes of manipulator)

- Uniform scale = **drag**
- Squish & stretch = **Shift-drag**

Rotate (drag green balls of manipulator)

- Constrained rotate = **drag**
- Free rotate = **Shift-drag**
- Move center of rotation = **Ctrl-drag**

Edit

- Undo = **Ctrl-z**
- Redo = **Shift-Ctrl-z**
- [Edit selected PEP object](#) (launches appropriate editor) = **Ctrl-e**
- Cut = **Ctrl-x**
- Copy = **Ctrl-c**
- Copy clone = **Shift-Ctrl-c**
- Paste copy = **Ctrl-v**
- Paste clone = **Shift-Ctrl-v**
- Delete = **Backspace**

PEP Move, Resize, and Rotate (drag handles of [PEP Jack](#))

- Position PEP Jack = **drag diamond**
- Constrain move PEP Jack in plane of green balls = **Shift-drag diamond**
- Constrain move PEP Jack along axis = **Ctrl-drag diamond**
- Move selected PEPs = **drag on any PEP**
- Constrain move selected PEPs = **Shift-drag on any PEP**
- Move selected PEPs in perpendicular direction = **Ctrl-drag on any PEP**
- Scale selected PEPs about center of PEP Jack = **drag end cubes**
- Scale selected PEPs along axis of PEP Jack = **Shift-drag end cubes**
- Scale selected PEPs in plane of green balls = **Ctrl-drag end cubes**
- Constrained rotation of PEPs about center of PEP Jack = **drag green balls**
- Free rotation of points about center of PEP Jack = **Shift-drag green balls**
- Snap movement of PEPs is affected by snap target, layout grids, and by the PEP snap options panels.

Animating Models (use the [Keyframe Animator](#))

- Open Animation = **Ctrl-o**
- New Animation = **Ctrl-n**
- Add Member = **Ctrl-a**
- Cut (keyframes) = **Ctrl-x**
- Copy (keyframes) = **Ctrl-c**
- Paste overlay at current time = **Ctrl-v**
- Delete keyframes = **Backspace**

- Undo* = ***Ctrl-z***
- Redo* = ***Ctrl-Shift-z***

***Note:** Undo and Redo in the keyframe animator acts on the latest change; it doesn't matter whether that change is in the keyframe animator or in the scene.

Wiring Routes ([Outline Editor](#))

- Create a route = **Click on two mating terminals**
- Remove a route = ***Ctrl-click* on terminal**
- Follow a route = ***Alt-click* on wire lead**
- View info on route = ***Shift-click* on wire lead**
- Create route (without clearing well) = ***Shift-click* on terminal**

Abbreviations

- ***PEP*** refers to points, edges, and polygons.
- ***Ctrl-Alt-u*** means to press at the same time the *Ctrl*, *Alt*, and *u* keys on your keyboard. Keys always appear in italics.
- **click** means to quickly press and release the left mouse button
- **click-middle** means to quickly press and release the middle mouse button
- **drag** means to press and hold down the left mouse button while dragging the mouse
- **drag-middle** means to press and hold down the middle mouse button while dragging the mouse

Hints

- When dragging on-screen manipulators, hold down *Shift* to add constraints to your dragging motion, or hold down *Ctrl* to perform alternate actions.
- At any time in the primary window, hold down *Alt* to enable the viewing controls.

Opening and Saving a Scene

on this page: [opening](#) | [saving](#) | [crash handling](#)

Opening a Scene

Choose *File > Open* and select a file in the file browser. You can open files created in the following formats: VRML 1.0, Inventor, VRML 2.0, any files created with Cosmo Worlds. If you install *xlators_3d* from the IRIX™ 6.*n* CD, you can also open files in these formats: Alias, DXF, Obj, SLA, Softimage.

Saving Changes

Choose *File > Save* or *Save As*

Note: The files that Cosmo Worlds saves are **not** always VRML 2.0 compliant. The nodes they contain are a superset of VRML 2.0 with enhanced authoring capabilities. Use *File > [Package](#)* to write out a fully compliant VRML 2.0 file.

Crash Handling

When Cosmo Worlds crashes, it saves two files in your current working directory:

- A file that may contain the scene immediately before the crash. For example: *cosmoworlds-crash.7667.wrl*

You can open this file if you had unsaved changes you want to recover.

- A detailed crash log. For example: *cosmoworlds-crashlog.7667*

Please include this file in any bug reports you send to Silicon Graphics; it contains information useful for determining why the crash occurred.

Steps for Packaging

Find it: *File > Package*

Steps

1. After you've saved the current file, select *File > Package*. The *Choose Package Directory* dialog appears. Specify a package directory. If it does not exist, it is created and the current file is added as a root document within that directory.

Note: Cosmo Package works only on saved files. Be sure to save any changes in Cosmo Create or Cosmo Worlds before packaging or restarting a package.

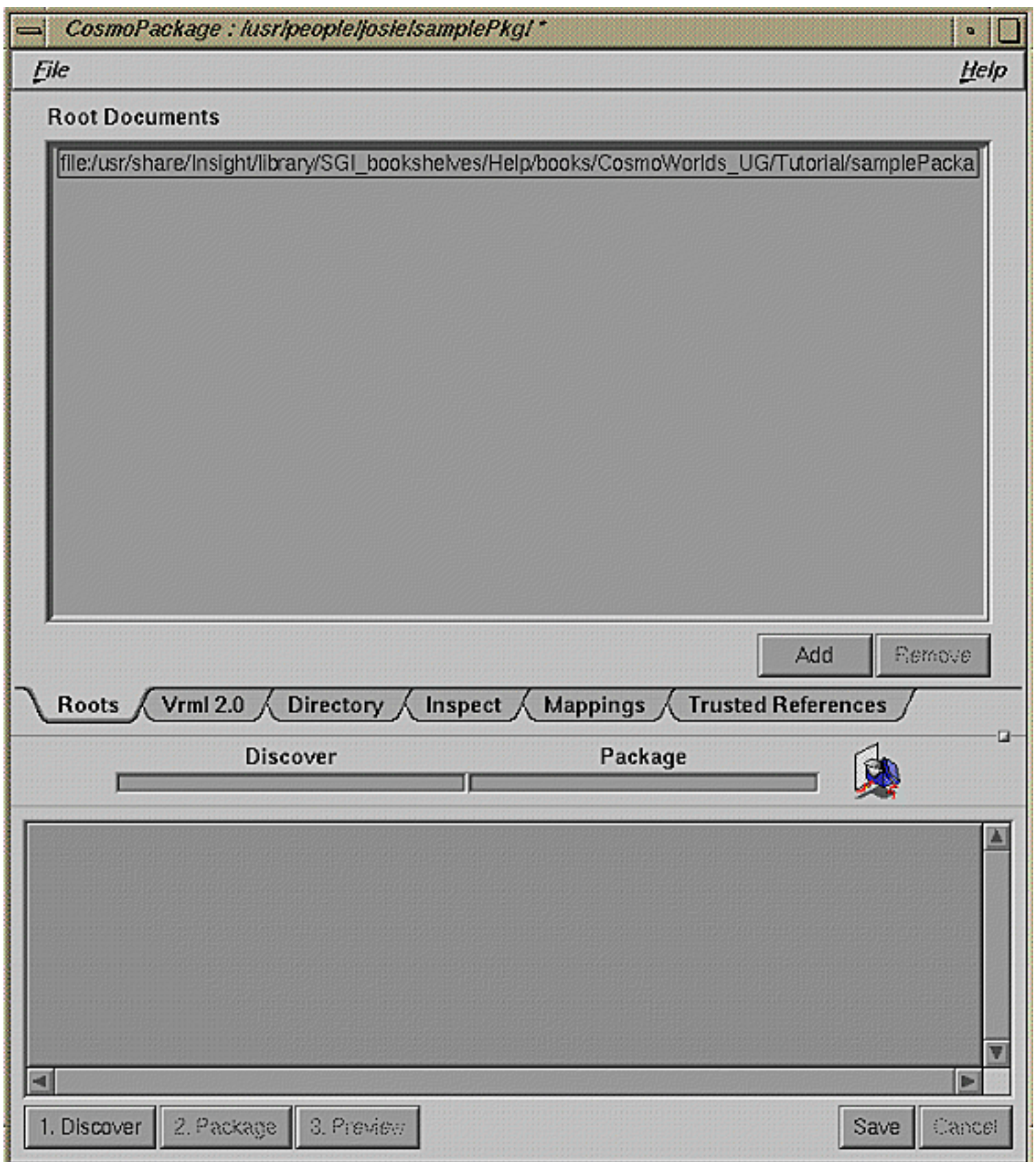
2. Choose *Roots > Add* and select the [root](#) to add if it isn't already there. Each root filename is expanded to a full URL beginning with *file:/.* The discovery stage starts with the root file (or files). To delete a root, select it and press *Remove*.
3. Set [VRML preferences](#) using the *Vrml 2.0* tab.
4. Specify [default index filenames](#), if any.
5. Specify [custom mappings](#), if any.
6. Specify [trusted references](#).
7. Click the [Discover](#) button to start the discovery stage. CosmoPackage finds all the local files referenced by the root file, as well as the local files referenced by those files. All local files referenced by URLs become part of the package for this document or scene. For VRML files, discovery includes such elements as inlines, textures, images, and sounds. For HTML files, it includes such elements as background images, embedded plug-in source code, and anchors. The discovered files scroll in the box at the bottom of the CosmoPackage window. The moving slider indicates the progress of the discovery stage. Note that the slider may move backwards during discovery as the packager finds new files that need to be added. Trusted references are ignored during discovery.
8. At any time, use the [Inspector](#) to view additional information about the documents being packaged.
9. Click the *Package* button to start the packaging stage.

This step creates the *stage/* subdirectory in the package directory, as well as an internal database used to store your packaging preferences. The packaged files scroll in the box at the bottom of the CosmoPackage window.

10. Click the *Preview* button to view the packaged document or scene. If your package has more

than one root, you are prompted to choose a root. This action brings up your preferred browser on the packaged version of the specified root. Or, choose *Preview Using* to specify a different browser to use for previewing the document. (Browsers are configured using the *File > Preview Using* option in Cosmo Create.)

11. Be sure to save the completed package so that it's written to disk. As with any document, when you save, you're overwriting the previous saved copy of the package.



Jump to:

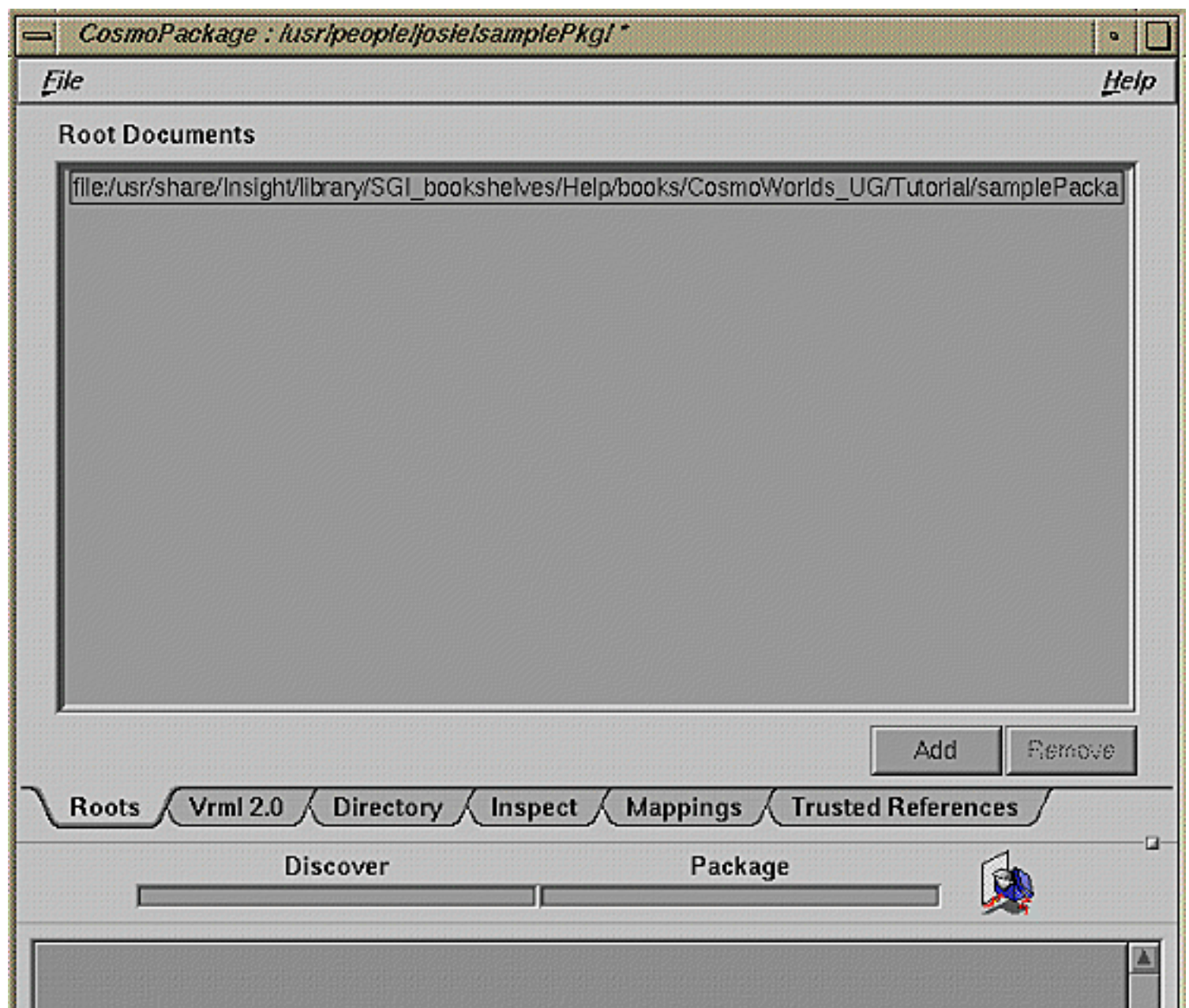
- [CosmoPackage Quick Reference](#)
- [Packaging a Document or Scene](#)
- [Possible Packaging Errors](#)

Specifying Root Documents for Packaging

A *root* document is a top-level file in the "tree" of related documents that make up your package. During the discovery stage, all documents referenced by this root document, as well as all documents referenced by those documents, are found and included as part of the package. Your document or scene can have multiple roots.

When you first start CosmoPackage, the current document is listed as a root in the package directory.

- Click the *Roots* tab to display the Roots panel, as shown below.
- Press the *Add* button to add root documents to the package directory. The *File Selection* dialog appears. Type the complete path for the file, or browse the current directories and select the path before you type the filename.
- To remove a root, highlight the root filename, then press *Remove*.





Jump to:

- [CosmoPackage Quick Reference](#)
- [Steps for Packaging](#)

CosmoPackage Quick Reference

The CosmoPackage utility allows you to specify the following things when you package a document or scene:

- [Root](#) documents or scenes of the package.
- [Vrml 2.0](#) preferences, such as amount of floating point precision and desired file formats. (You can skip this panel if your package does not contain any VRML files. This panel is present only if you have Cosmo Worlds installed on your system.)
- A default [directory index file](#) to be packaged in place of *index.html* whenever a directory reference occurs during packaging.
- [Custom mappings](#) that specify a new location for certain files when they are packaged.
- [Trusted references](#) that do not need to be discovered or packaged.

In addition, CosmoPackage allows you to [inspect](#) the details concerning the packaged documents at any time during packaging.

Jump to: [Steps for Packaging](#)

Specifying VRML Preferences

on this page: [floating point precision](#) | [.wrl suffix](#) | [gzip](#) | [optimizing textures](#) | [VRML 2.0 file formats](#) | [imgcopy and dmconvert utilities](#)

Click the *Vrml 2.0* tab to display the VRML 2.0 Preferences panel of CosmoPackage. This panel is present only if you have Cosmo Worlds installed on your system. You can set preferences only before the discovery stage. If you need to change any preferences after discovery, you'll need to restart the packaging process.

Floating Point Precision

The *Floating Point Precision* pulldown menu allows you to specify the maximum number of significant digits to be used for the decimal part of floating point values in VRML files. Decimal values are rounded to the specified number of digits. Since shorter files download more quickly, you want to specify the minimum floating point precision necessary to render your scene accurately.

The default is to write out all significant digits that are stored for each value.

Adding the .wrl Suffix

By default, CosmoPackage adds a *.wrl* suffix to all VRML files that don't already have one. Note that Netscape does not recognize a file as VRML without the *.wrl* extension. To disable this feature, click the box to remove the check.

Compressing Files with gzip

By default, VRML files are compressed using *gzip* before they are packaged. Compressed files download faster and use less disk space, so use of this feature is recommended. Most browsers automatically decompress files that have been compressed. To disable this feature, click the box to remove the check.

Optimizing Textures

Check the *Optimize Textures* box if you want CosmoPackage to optimize textures automatically. When this feature is enabled, CosmoPackage performs the following optimizations:

- It looks for textures that are used multiple times in the same file and creates instances for all but the first use of a given texture.
- It removes object texture coordinates, texture transforms, and texture coordinate indices if there is no texture associated with the object. (These nodes are added when an object is

converted to PEP, but they may be unused.)

- It removes texture coordinate indices if they are the same as the coordinate indices. (In VRML, if no texture coordinates are supplied, the coordinate indices are used instead, so there's no need for the duplication.)

If a texture node has routes connected to it, no optimizations are performed.

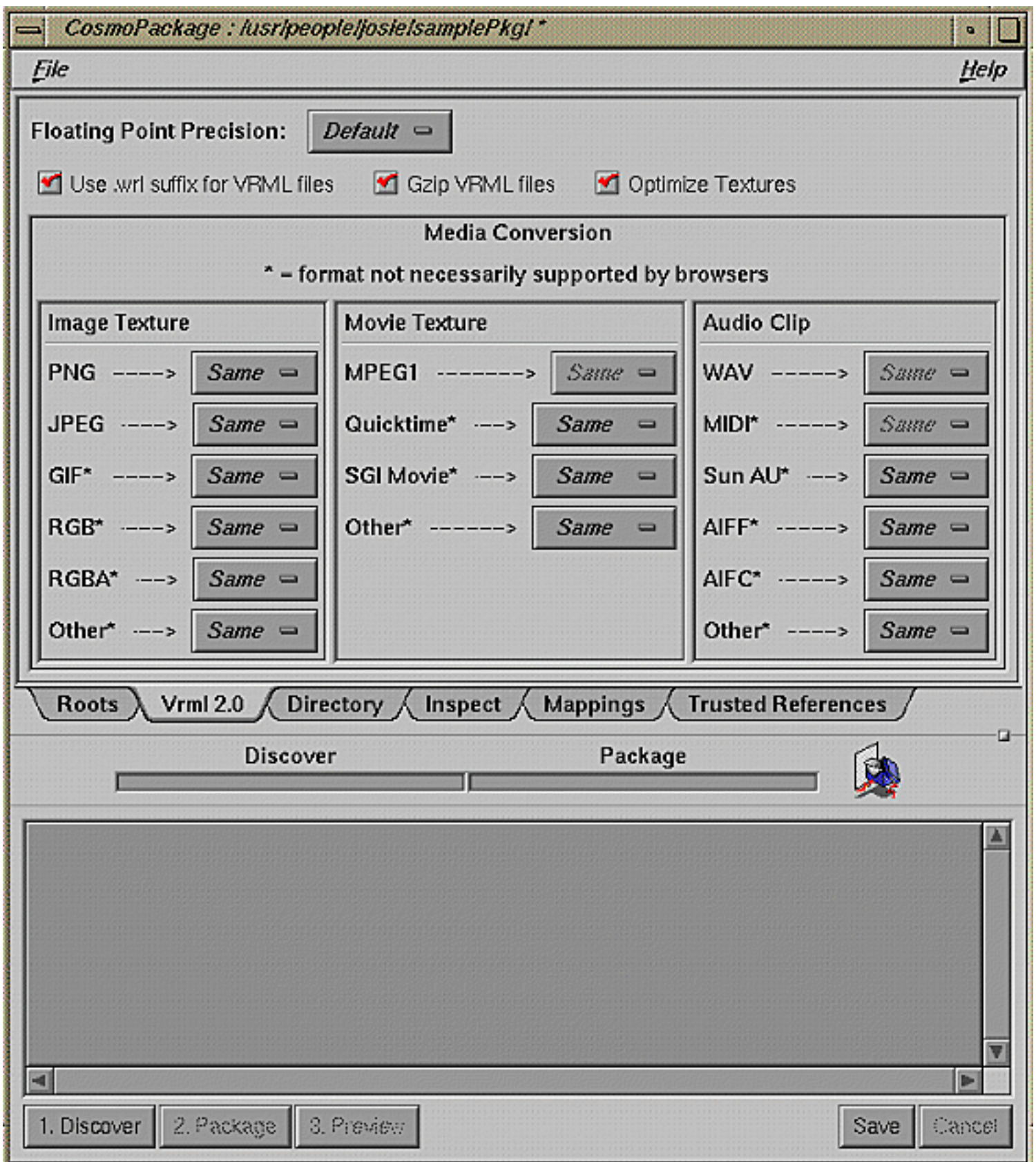
VRML 2.0 File Formats

The Media Conversion panel allows you to specify the file format of your packaged images, movie textures, and audio clips. Since browsers are not required to support all possible formats, you'll want to convert your files to the formats required by the VRML 2.0 Specification. Note that this panel operates only on local files; remote URLs and trusted references are untouched. By default, no file conversion occurs.

In this panel, an asterisk next to the file format indicates that this format may not be supported by all browsers. The VRML 2.0 Specification requires support for PNG and JPEG image formats, MPEG1 movie texture format, MPEG1 audio clip format, and WAV audio format. Support for the GIF image format and MIDI file type 1 audio format is recommended but not required.

Obtaining Conversion Utilities

The utility used for image conversion is *imgcopy*. The utility used for converting movies and sounds is *dmconvert*. These utilities are not a standard part of your operating system. If they are not installed, an error message will appear on your screen. Obtain the utilities from the CD or Web site where you obtained Cosmo Worlds, if they are not already on your system.



Jump to:

- [CosmoPackage Quick Reference](#)
- [Steps for Packaging](#)

Specifying Default Directory Index Filenames

When a file contains a reference to a directory, Cosmo Package packages the *index.html* file within that directory, if it finds one. If you have specified a different default filename to be loaded when a directory is referenced on your system, you can enter that filename in the *Directory* panel in Cosmo Package.

You can enter any number of default index filenames. During discovery, Cosmo Package loads the first default index filename it finds in the specified directory.

Jump to:

- [Packaging a Document or Scene](#)
- [CosmoPackage Quick Reference](#)

Packaging a Document or Scene

Find it: *File > Package*

Before you publish your documents or scenes on the Web, you need to package them using CosmoPackage. *Packaging* refers to the process of locating all the files necessary to create a document, scene, or world and organizing the files for publication on a server. *Publishing* refers to the process of copying the packaged files to a Web server where they can be accessed by Web clients.

Packaging accomplishes a number of things:

- It ***collects all the files*** needed to make up your document or world on the Web and copies them into a single directory. As it copies the files, it ***converts absolute paths*** to relative paths so that the package can be moved to a different location (for example, a Web server).
- For VRML files, it ***writes out a modified version of the VRML file format*** so that it conforms to the current VRML standard. Files saved in Cosmo Worlds can be viewed directly using Cosmo Player on a system that has Cosmo Worlds installed on it, but the files need to be packaged before they are published on the Web.

Note: Packaging is a one-way process. It strips authoring information and optimizes the files. Without this stripped information, you may not be able to edit embedded images and plug-in applications in an HTML file or animations in a VRML file. For this reason, if you want to continue to edit a file, work on the file you originally saved in Cosmo Create or Cosmo Worlds. Package the file again when you've finished editing it.

Major Stages

When you select *File > Package*, the CosmoPackage utility appears. The three buttons in the lower left corner allow you to step through each stage of the packaging process:

Discover

This stage finds all the local files needed to support your documents. All local URLs referred to in your file (for example, inlines, textures, sounds, and images) are located and included as part of the package for this file. Remote files and [trusted references](#) are ignored during this stage.

Package

This stage copies the files required for your package into a single staging directory. Absolute paths are changed to relative paths so that the world can be published on a Web server. During the packaging stage, CosmoPackage also optimizes VRML files and may rename them (for example, by adding a .gz or .wrl suffix). SGI-specific attributes are stripped from HTML files.

Preview

This stage allows you to view the packaged files so that you can test links, animations, sensors, scripts, navigation through VRML worlds, and layout of HTML pages.

Be sure to check the icon in the lower-right portion of the CosmoPackage window. If the package resembles an open box, packaging is still in progress (you're in the discovery or packaging stage). If the icon is a closed box with a tied ribbon, packaging is complete, and you have successfully completed both the discovery and packaging stages.

Note: Remember, just as with any document, you need to save the package in order for it to be recorded on disk.

CosmoPackage works with files in the following formats:

- **Cosmo Worlds** (these files are converted to VRML 2.0 format; Cosmo Worlds files contain some additional features that are not strictly part of VRML 2.0)
- **VRML 1.0** and **VRML 2.0** (VRML 1.0 files are converted to VRML 2.0)
- **Inventor** (Inventor files are converted to VRML 2.0)
- **Cosmo Create** (These files contain some SGI-specific data that is not part of standard HTML; although most browsers simply ignore the extra data, the packaging stage removes it.)
- **HTML**

Other document types (such as .rgb and .gif image types, Java class files, and Shockwave plug-in files) are not searched for references to other local documents, they do not have their references corrected, nor are they optimized; they are just moved to the packaging directory.

Jump to:

- [CosmoPackage Quick Reference](#)
- [Steps for Packaging](#)
- [Possible Packaging Errors](#)

Specifying Trusted References

Click the *Trusted References* tab to display this CosmoPackage panel.

One of the useful features provided by CosmoPackage is that it interprets each file reference contained in the packaged files, searches for it, and returns an error if the file isn't found. The Trusted References feature overrides this protection and is thus not recommended for casual use. However, if you are sure that a certain file or group of files will be in a specific location on the server when the package is published and you don't want CosmoPackage to verify its presence, this feature is for you.

Press the *Add* button to add a trusted reference to the package. Enter a directory name, a complete filename, or a URL into the text box. If you specify a directory, everything under that directory will be considered a trusted reference.

Jump to:

- [CosmoPackage Quick Reference](#)
- [Steps for Packaging](#)

Possible Packaging Errors

If you encounter an error during discovery or packaging, a dialog box appears notifying you of the problem.

Possible errors during discovery include:

- The specified root is not found.
- A referenced URL points to a nonexistent file. (If you know that a referenced file will be on the server, you can use the Trusted References option to skip the discovery stage for that file.)
- The file contents are not in the expected format (for example, VRML, HTML).
- The files cannot be converted. Check to be sure that you have the current version of the *imgcopy* and *dmconvert* utilities installed.

Possible errors during packaging include:

- You are out of disk space.
- Two files cannot be mapped to the same location. For example, you might have two files named *brick.gif* in different paths. If they need to be moved to the same directory (in the packaging stage), one needs to be renamed. Selecting the *Auto Handle Overlapping Names* box in the *Mappings* panel allows CosmoPackage to rename these files for you (you'll end up with *brick.gif* and *brick__2.gif*). (An alternative is to create a custom mapping for one of the duplicate files so that the files aren't mapped to the same directory.)

Jump to:

- [Previewing and Packaging the Scene](#)
- [Steps for Packaging](#)

Specifying Custom Mappings

A *mapping* specifies the placement of a file in the staging area. In the screen below, for example, all files found in the directory */hosts/manny.engr.sgi.com/disk2* will be placed in the *vrml/* subdirectory of the package directory's stage. CosmoPackage allows you to specify a general default mapping strategy as well as specific custom mappings for packaged files.

Click the *Mappings* tab to display the Mappings panel of CosmoPackage. Click the *Inspect* tab to view mappings for individual files as they are packaged.

Custom Mappings

You can specify your own mappings for local files, remote files, and trusted references. This feature allows you to create a cleaner, more logical file structure for the packaged files than their original structure.

In the *Map From* field, specify the location of the files in the source directory. In the *Map To* field, specify the desired location of the packaged files in the *stage/* subdirectory of the package directory.

If you try to map two files to the same location, an error occurs. To allow the system to handle such errors, check the *Auto Handle Overlapping Names* box (the default). To disable this feature, click the box to remove the check. When this feature is enabled, the system adds an extra digit to the second name to prevent the duplication.

Default Mapping

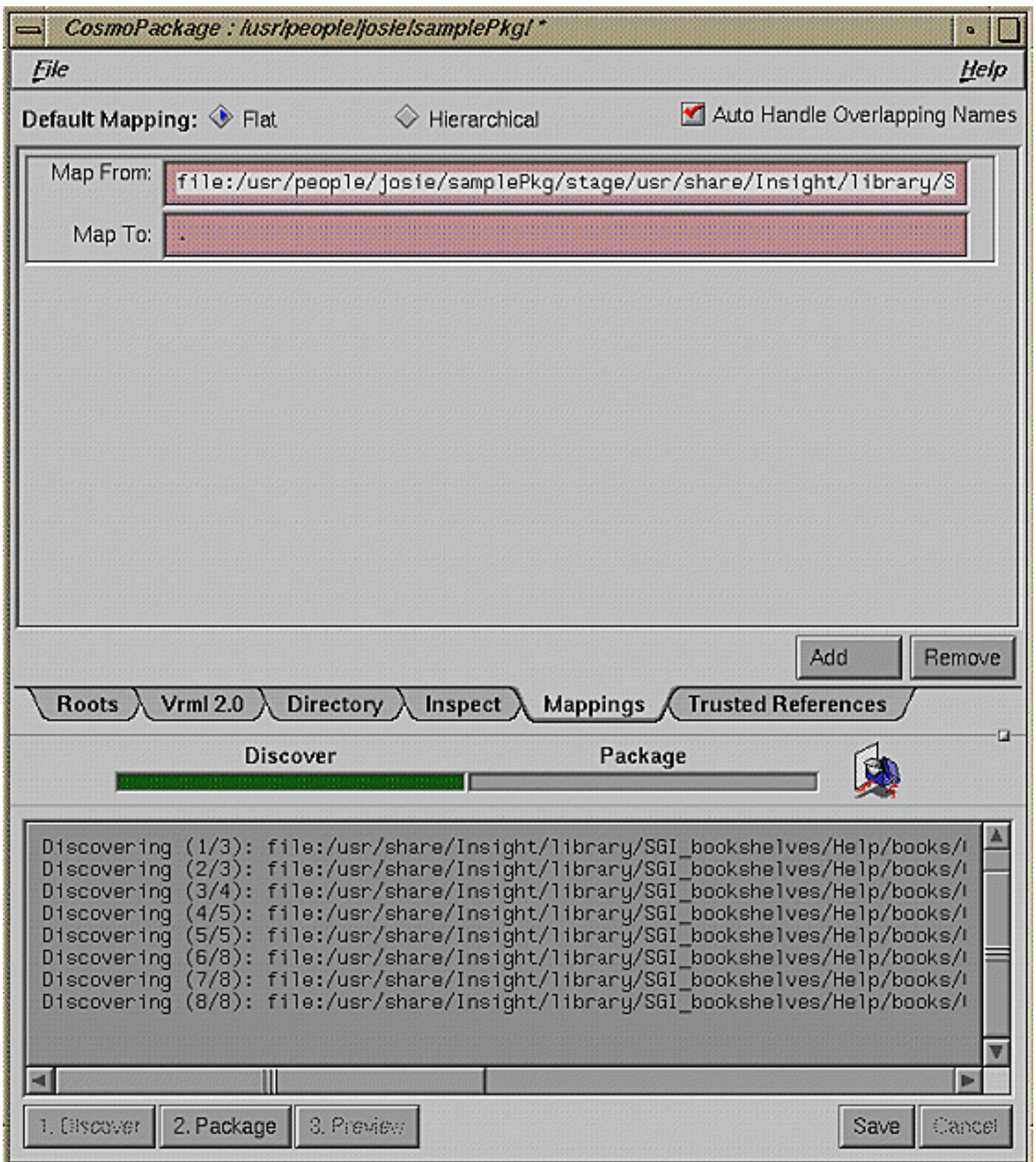
If no custom mapping is specified for a given document, a default mapping strategy is used. You can choose between one of two general techniques for mapping files from the source directory on the authoring side to the packaged directory:

Flat

This choice (the default) maps all files directly to the staging area of the package directory, regardless of where they are in the file hierarchy on your system. The staging area is a subdirectory (*stage/*) in the package directory. The result is a completely flat file structure.

Hierarchical

This choice preserves the file hierarchy found on the author's system. It moves the necessary files to the *stage/* directory, mirroring the author's file structure.



Jump to:

- [CosmoPackage Quick Reference](#)
- [Steps for Packaging](#)
- [Try It! Packaging a Sample World](#)

Try It! Packaging a Sample World

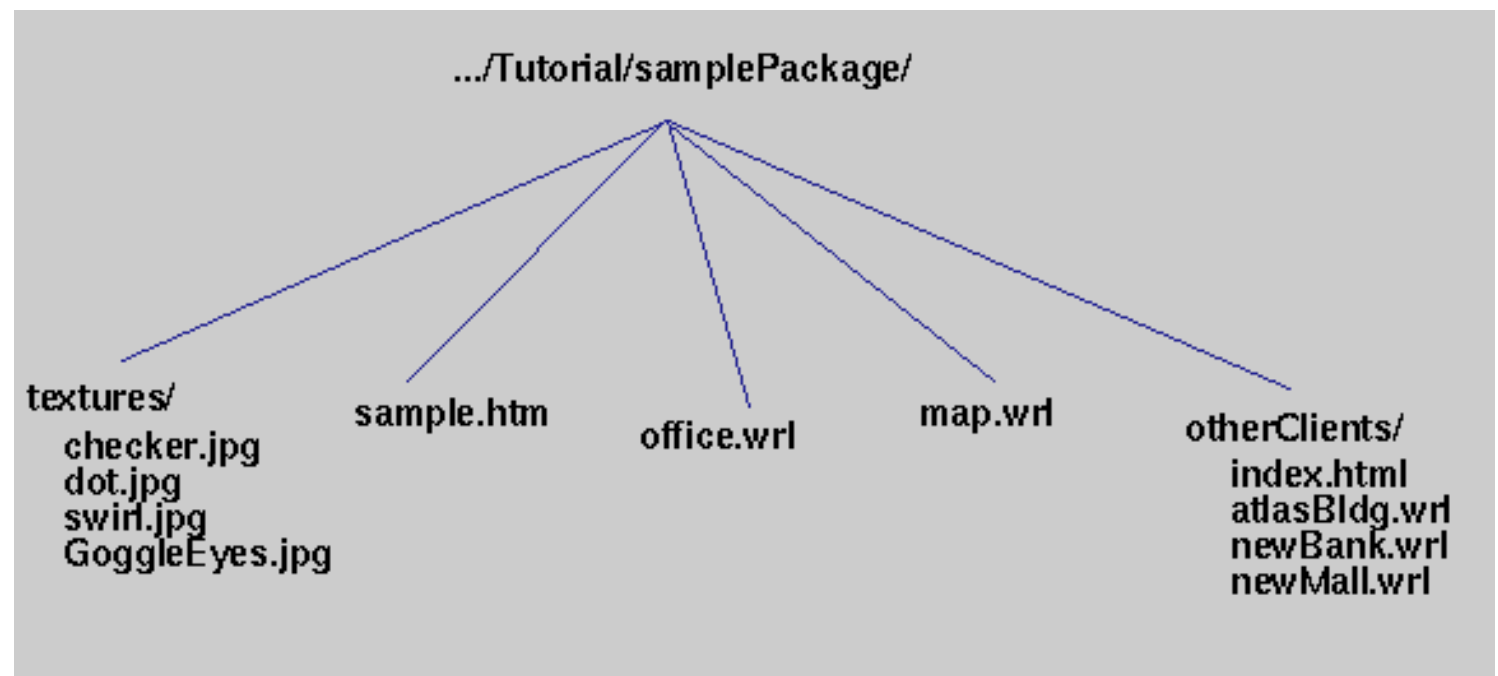
on this page: [defaults](#) | [directory structure](#) | [mappings](#) | [directory index](#) | [trusted references](#)

After you've read some of the background material on CosmoPackage, try experimenting with this sample set of files to see how you can set various options and what happens during packaging. For complete information, be sure to read all the Help cards listed at the end of this section.

The sample files are located in this directory:

/usr/share/Insight/library/SGI_bookshelves/Help/books/CosmoCreate_UG/Tutorial/samplePackage

This directory contains the following files and subdirectories:



Note to Cosmo Create users: The *.wrl* suffix is for files created in Cosmo Worlds. The *.jpg* suffix indicates an image file in JPEG format. If you do not have Cosmo Worlds installed, *.wrl* files will be treated as "black boxes"; they will be copied as is and will not be processed (their references will not be discovered and they will not be converted to standard VRML).

Using the Default Values

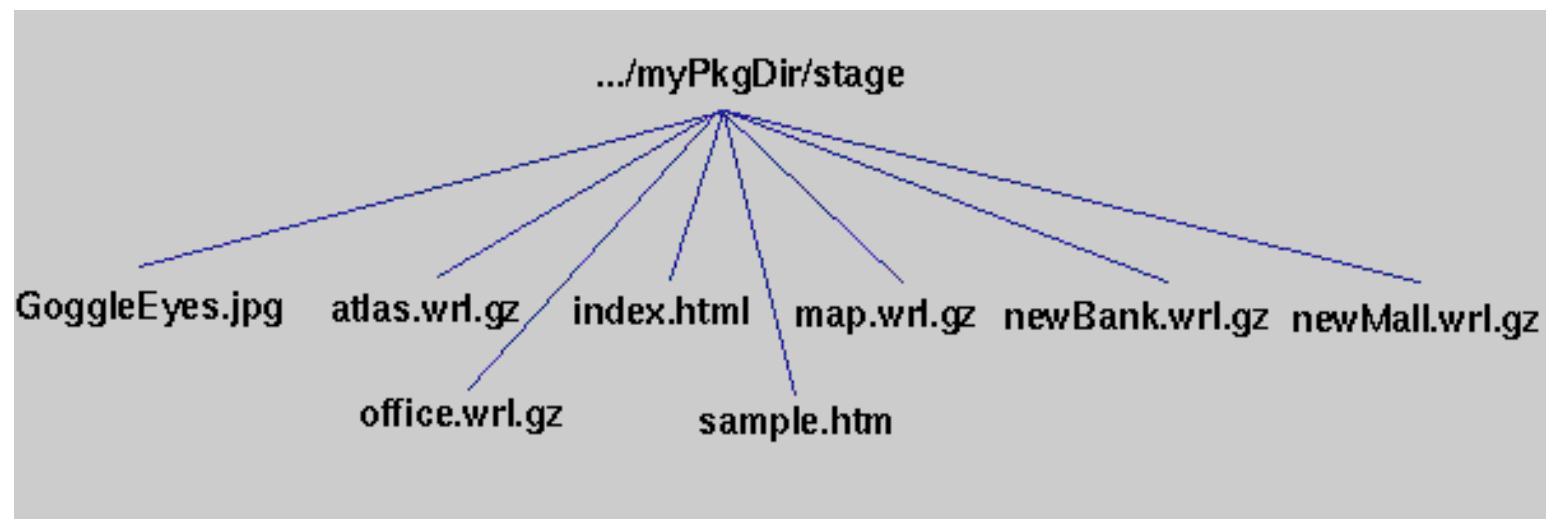
First, try running CosmoPackage using the default settings. Use the file *sample.htm* as the root file of your package. This file contains links to *office.wrl*, *map.wrl*, and the *otherClients/* directory. The *map.wrl* file uses the *GoggleEyes.jpg* texture from the *textures/* directory.

By default, when a file refers to a directory, CosmoPackage packages the *index.html* file within that directory, if there is one. See [Referring to a Directory Index](#). In the example, the file *index.html* contains links to the other three *.wrl* files in the *otherClients/* directory.

Follow these steps to package *sample.htm*:

1. Open *sample.htm* and choose *File > Package*.
2. Choose a package directory. For example, type *myPkgDir* in the text box and click *OK*.
The file *sample.htm* is added as the root of this package.
3. Click the *Discover* button. CosmoPackage looks for all files linked to the root file and adds them to the package.
4. Click the *Package* button. CosmoPackage creates the package containing the necessary files.
5. Click *Preview* to test out the links in your package.
6. Click *Save*. The package is saved in the directory */myPkgDir/stage* in your working directory.

Now examine the contents of the *stage/* directory. You'll see that it looks like this:



Important things to notice at this point are:

- The discovery search is recursive; each root document or world is included in the package, as well as any other documents or worlds included as links in the "parent" document. Each linked document, in turn, is searched for other documents included as links. The package contains only the files to which the root file *sample.htm* (or any of its "children") contain links. Several of the textures in the sample directory are not referenced by any documents or worlds, and so they're not included in the package.
- The directory structure is completely flat. The *textures/* and *otherClients/* directories were not created in the *stage/* directory.
- The VRML files (ending in *.wrl*) were compressed using the gzip utility, and the *.gz* suffix was added to the filenames.

Maintaining Your Directory Structure

If you have a lot of files that are carefully arranged into logical subdirectories, you may not want your package to have a completely flat file structure. Follow these steps to preserve a hierarchical file structure:

1. Click *Restart*.
2. Click the *Mappings* tab. Then click the *Hierarchical* check box.
3. Click *Discover*.
4. Click *Package*.
5. Click *Save*.

Now check the contents of `/workingDirectory/myPkgDir/stage`. You'll find the *usr/* subdirectory, and if you dig deep enough, you'll finally find the contents of your package in the following directory:

```
/workingDirectory/myPkgDir/stage/usr/share/Insight/library/SGL_bookshelves/Help/books/
CosmoCreate_UG/Tutorial/samplePackage
```

Clearly, you need a way to shorten this path! The answer is to add a *custom mapping*, as described in the following section.

Directory Mappings

Follow these steps to map files from your original working directories to the package directory.

1. Click *Restart*.
2. Click the *Mappings* tab. Be sure the *Hierarchical* box is still checked.
3. Click the *Add* button. In the *Map From* field, specify (all on one line):

```
file:/usr/share/Insight/library/SGL_bookshelves/Help/books/
CosmoCreate_UG/Tutorial/samplePackage/
```

Be sure to include the *file:/* specification at the beginning.

In the *Map To* field, specify

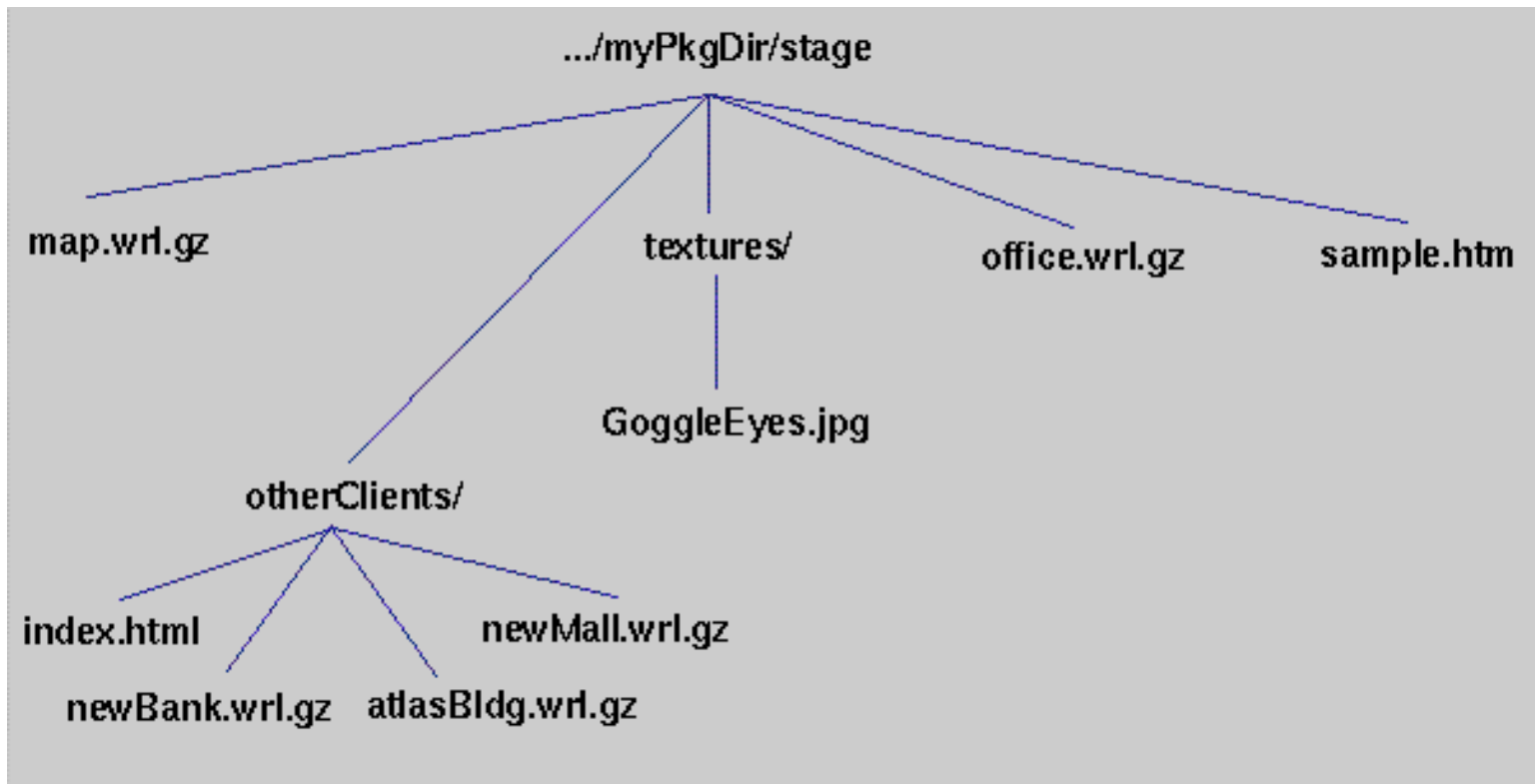
.

(to indicate the *stage/* directory)

4. Click *Discover*.
5. Click *Package*.

6. Click *Save*.

Now, you have a short, readable path to your files, and your original directory structure is maintained (with the *otherClients/* and *textures/* directories as part of your package):



Referring to a Directory Index (index.html)

This example illustrates another technique you may find useful. The *sample.htm* file contains a link to the *otherClients/* directory. By default, the *index.html* file is packaged when an HTML link specifies a directory. In the example provided, the *index.html* file, in turn, contains references to all the files in that subdirectory, which causes them to be pulled recursively into the package.

You can also specify another filename for the packager to use when it receives a directory reference. If you want CosmoPackage to use a filename other than *index.html* when it encounters a directory reference, add that name to the *Directory* panel in CosmoPackage.

Trusted References

In some cases, you know that certain files will already be placed on the server, and you don't want them included in your package. This is where trusted references are useful.

To test their use, assume that all textures are currently placed on the server in a specified location. To omit the textures from your package, follow these steps:

1. Click *Restart*.
2. Click the *Trusted References* tab.

3. Click *Add*.
4. Type the path of the trusted reference (all on one line):

***file:/usr/share/Insight/library/SGL_bookshelves/Help/books/
CosmoCreate_UG/Tutorial/samplePackage/textures/***

Be sure to include the *file:/* specification at the beginning of the trusted reference.

5. Click *Discover*.
6. Click *Package*.
7. Click *Save*.

Now, if you examine the contents of the package directory, you'll notice that the textures are missing, since you've assumed that they are already on the server. CosmoPackage did not follow any links to the *textures/* directory.

Tip: If your files contain relative references, such as *../textures*, you also need to specify *../textures* as a trusted reference.

Jump to:

- [Packaging a Document or Scene](#)
- [CosmoPackage Quick Reference](#)

Links Followed During Discovery

This page lists the HTML and VRML links that are followed during the discovery stage of packaging.

The following HTML links are followed during discovery:

Element	Attribute
A	HREF
APPLET	CODE
AREA	HREF
BGSOUND	SRC
BODY	BACKGROUND
EMBED	SRC
FIG	IMAGEMAP
FIG	SRC
FORM	ACTION
FRAME	SRC
HR	SRC
IMG	SRC
IMG	LOWSRC
IMG	USEMAP
INPUT	SRC
ISINDEX	ACTION
LI	SRC
LINK	HREF
LINK	SRC
NOTE	SRC
OVERLAY	IMAGEMAP

OVERLAY	SRC
PARAM	VALUE (if VALUE = REF or OBJECT)
SCRIPT	SRC
STYLE	HREF
SOUND	SRC
TABLE	BACKGROUND
UL	SRC

The following VRML links are followed during discovery:

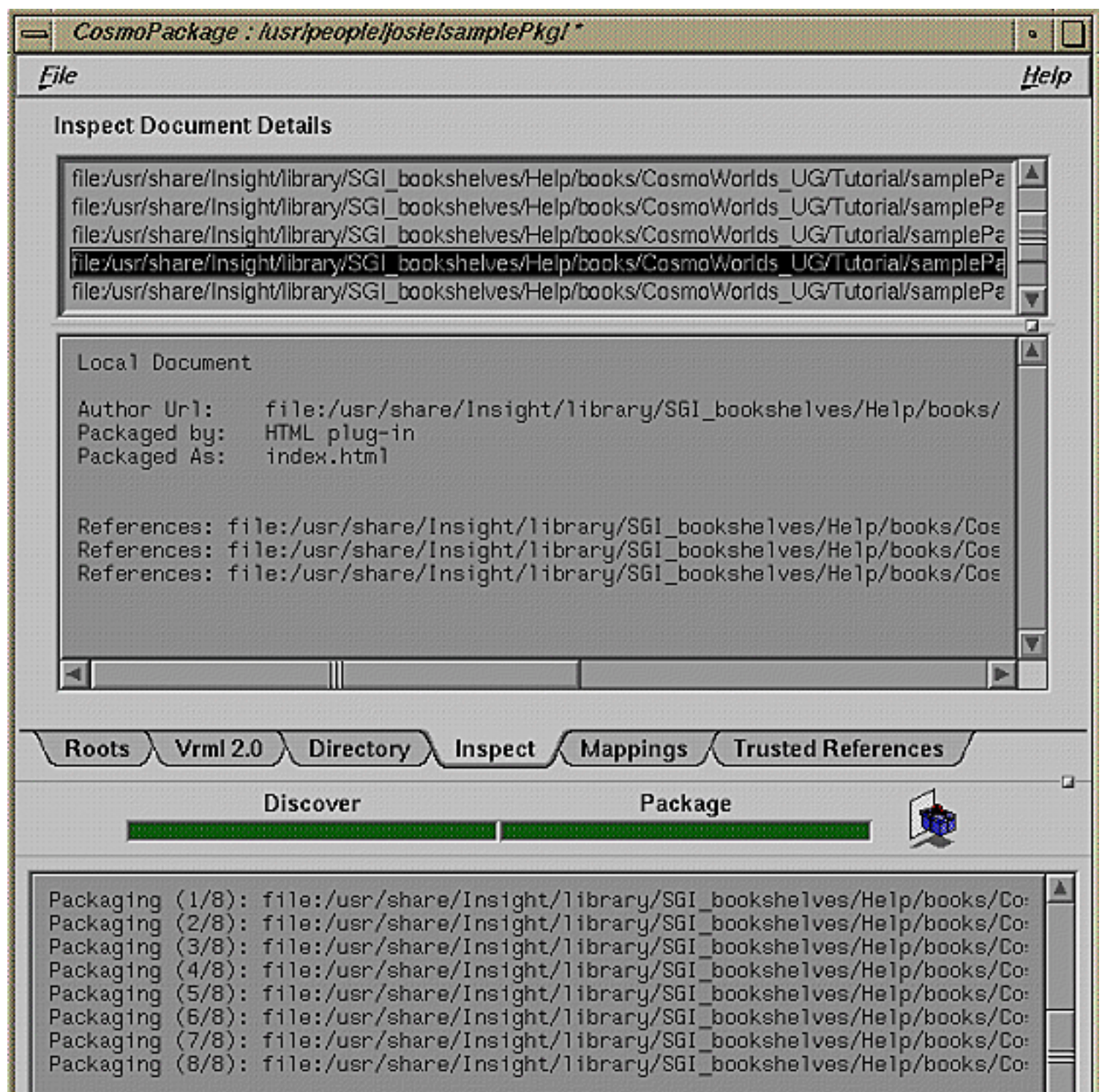
Node	Field
Anchor	url
AudioClip	url
Background	backUrl
Background	bottomUrl
Background	frontUrl
Background	leftUrl
Background	rightUrl
Background	topUrl
Inline	url
ImageTexture	url
MovieTexture	url
Script	url
ExternProtoDef	url

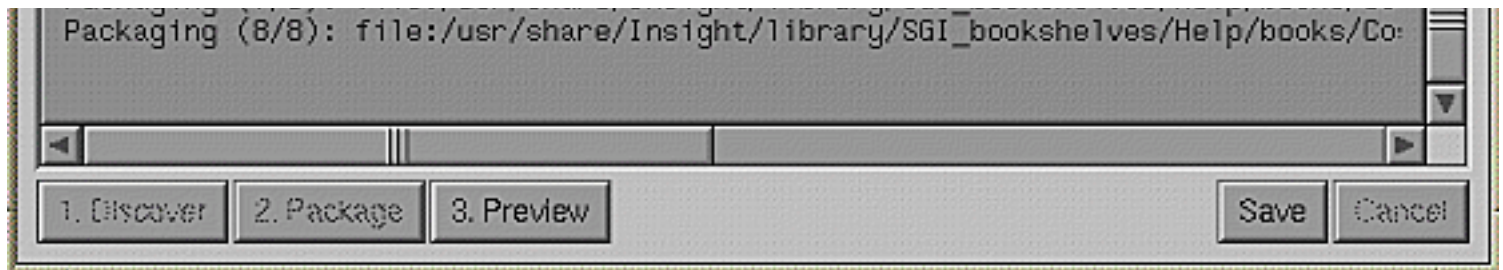
Jump to: [CosmoPackage Quick Reference](#)

Inspecting the Packaged Files

Click the *Inspect* tab to display this CosmoPackage panel. Use this panel to inspect details concerning the packaged files. More information appears in the panel during each stage of packaging.

Highlight the name of the file you're interested in. This panel shows the location of each file on the authoring side (*Author Url*) and the mapping used for placing the file in the package directory (*Mapped Using*). The *Packaged As* entry shows the complete path and filename for the file in the package directory. The *Packaged By* entry shows the name of the plug-in application that packaged the file--for example, HTML, VRML, or Black Box (indicates copied, not interpreted). This panel also shows the inline objects and links (references) contained in the selected file.





Jump to:

- [CosmoPackage Quick Reference](#)
- [Steps for Packaging](#)

Viewing Objects and Scenes

on this page: [the basics](#) | [a quick trip to the cinema](#)

The Basics



Changing your view is an integral part of working with objects in a scene. Here are the basics:

- To view, switch to view mode by selecting the hand icon from the [Viewing toolbar](#).

You can also hold down *Alt* to temporarily toggle to View mode, or press *Esc* to switch between pick and view modes.

- Use the [Examiner Viewer](#) when you are looking at objects in your scene. Use the [Walk Viewer](#) to navigate through your scene.

Note: Editing tools which have their own windows to preview objects also use the Examiner and Walk viewers. Use shortcuts, or [turn on window decoration](#) in the editor.

- Switch between [free views](#) and [construction views](#). Construction views constrain the camera to a single plane and move the camera from [perspective projection](#)  to an [orthographic projection](#) .
- Use buttons on the Viewing toolbar and [Viewing palette](#) to [quickly set the home viewpoint and return to the home view](#).

Note: Changing your viewpoint of an object in the scene is different from [moving the object](#). Don't move an object by selecting it and dragging its manipulators when you really mean to [dolly, pan, or rotate](#) in the [Examiner Viewer](#), or navigate and tilt using the [Walk Viewer](#).

A Quick Trip to the Cinema

Cosmo Worlds uses a cinematography model to explain the way you look at objects in your scene. For example, when you rotate an object to change your view, you are moving the camera around the object (not moving the object itself). Here are some basic terms used to describe viewing:

- **Camera**--When looking at objects or navigating through your scene, imagine that an unseen camera is defining your view.
- **Dolly In or Out**--Slide the camera closer or further away from objects. Because the camera is sliding toward an object, you can dolly past the object to the other side of it.

To dolly, use the dolly thumbwheel in the lower right corner of the main window. In the Examiner Viewer, you can also place the mouse cursor over any part of the scene and *Ctrl-Alt*-drag middle up and down.

- **Zoom In or Out**--Here, the camera does not travel to or past objects in the scene. The camera "lens" magnifies the object.

In most cases, you will want to use dolly instead of zoom. To use zoom, hold the mouse cursor over the scene and press the right mouse button. From the menu that appears, choose Preferences. The Preferences Panel appears. Use the zoom slider here to zoom in or out.

- **Pan**--Pan means to slide the camera back and forth or up and down.

To pan, use *Ctrl-Alt*-drag (drag in the direction you wish to move the scene).

- **Tilt**--(available only in the Walk viewer) When you tilt the camera, you rotate it up or down, moving your view in the same way that you look up or down by tilting your head.

To tilt, switch to the Walk Viewer (*View > Walk Viewer* or *Ctrl-k*) and use the Tilt thumbwheel in the lower left corner of the main window.

- **Rotate**--(available only in the Examiner viewer) Use the *Rot X* and *Rot Y* thumbwheels at the lower left corner of the main window to rotate the camera around the center of the scene.

Jump to:

- [Tools to Use: Viewing](#)
- [Defining Viewpoints](#)
- [Keys & Shortcuts for Viewing](#)

Changing the Draw Style

You can change the way Cosmo Worlds renders your scene when you view it as a still or when you move objects. You can also set how Cosmo Worlds makes use of colors by changing the buffering style.

- [Draw style for static view](#)--This is the rendering style used when the camera is not moving. Change if your scene redraws too slowly when you move between different views.
- [Draw style for moving view](#)--This is the rendering style used when the camera is moving. Change if your objects update too slowly when you move them
- [Buffering style](#)--The default is double buffering. Change to single buffering for the best color resolution when monitor flickering is not an issue.

Jump to:

- [Viewing Objects and Scenes](#)
- [Tools to Use: Viewing](#)

Changing the Draw Style for a Static View

If objects in your scene take too long to render each time you change your view, you can change the draw style to one that is quicker to redraw.

In the viewer, click and hold right mouse button to bring up the popup menu. Choose *Draw Style* >

- *hidden line*--draws as wireframe, but removes lines not visible to current viewpoint
- *no texture*--shows model without texture
- *low resolution*--shows model at lower resolution or complexity
- *wire frame*--shows model as wire frame only
- *points*--shows only the vertex points making up each object
- *bounding box*--shows bounding box outline only for each object

Jump to:

- [Viewing Objects and Scenes](#)
- [Changing the Draw Style](#) (overview)
- [Changing the Draw Style for Moving Objects](#)
- [Changing the Draw Style Using Buffering](#)

Changing the Draw Style for a Moving View

If objects in your scene take too long to render while you are moving the camera, you can change the draw style to one that is quicker.

Find it: In the viewer, click and hold right mouse button to bring up the popup menu. Choose *Draw Style* >

- *move same as still*--object is rendered same as for static view
- *move no texture* --default, eliminates texture
- *move low res* --changes object to low resolution
- *move wireframe*--changes object to wireframe
- *move low res wireframe*--changes object to a low resolution wireframe with no depth information
- *move points*--changes object to points only
- *move low res points*--changes object to low-resolution points
- *move bounding box* --replaces each object with a bounding box

Jump to:

- [Viewing Tools to Use](#)
- [Changing the Draw Style](#)

Changing the Draw Style using Buffering

The buffering menu items let you control how Cosmo Worlds makes use of colors. By default, Cosmo Worlds is set to use *double buffer*, in which the color planes are split in half to prevent flickering when you move your view. This reduces the color resolution of shaded objects. *Single buffer* maximizes color resolution at the expense of visible flickering when the scene is being redrawn. *Interactive buffer* is a compromise, where the scene is drawn single buffered when the camera is not moving, and in double buffered mode when moving. Some color flashing will be noticeable when camera motion begins and ends.

Find it: In the [Examiner Viewer](#) or [Walk Viewer](#), click and hold right mouse button to bring up a menu. Choose *Draw Style* >

- *single buffer*--use for best color resolution (for example, when taking screen snapshots), screen flickers when you move the camera
- *double buffer*--default, prevents screen flicker when moving the camera
- *interactive buffer*--uses single buffer for static view, double buffer for moving view

Jump to:

- [Viewing Tools to Use](#)
- [Changing the Draw Style](#)

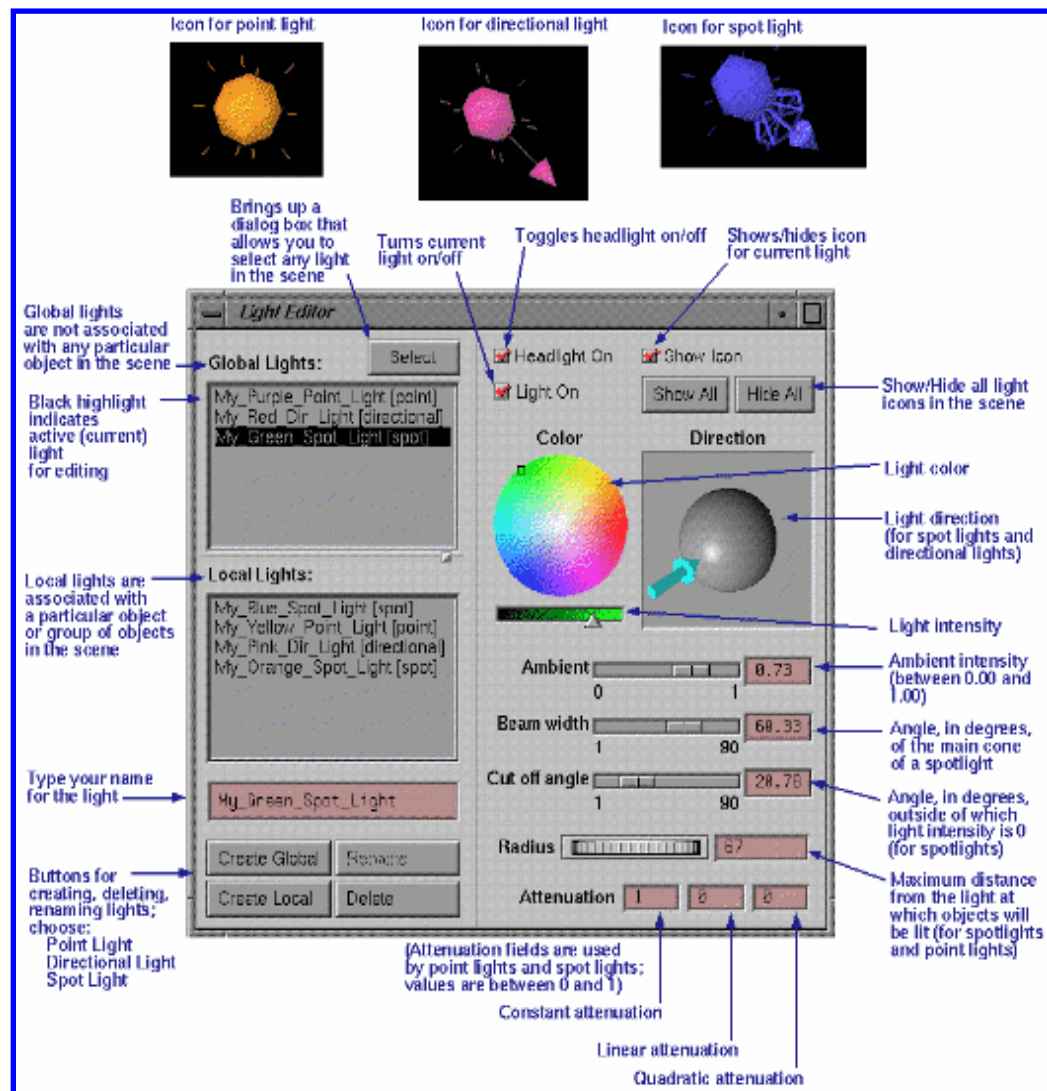
Light Editor



Find it: Click its button on the *Looks* palette:

Use to: Create lights; set and modify light attributes

How it works:

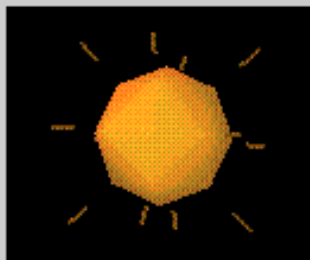


Click image for full view.

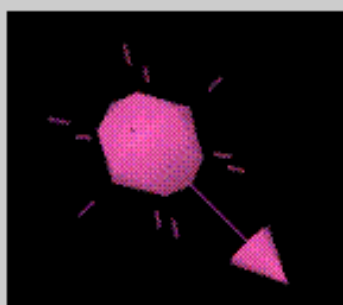
Jump to:

- [Creating and Editing Lights](#)
- [Light Types: What's the Difference?](#)
- [More about Spot Lights](#)
- [Tips for Efficient Lighting](#)

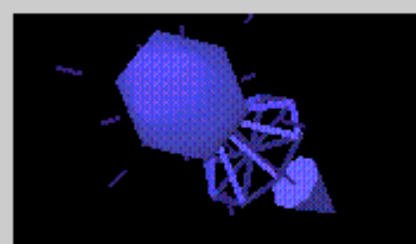
Icon for point light



Icon for directional light



Icon for spot light



Brings up a dialog box that allows you to select any light in the scene

Turns current light on/off

Toggles headlight on/off

Shows/hides icon for current light

Global lights are not associated with any particular object in the scene

Global Lights:

Select

My_Purple_Point_Light [point]
My_Red_Dir_Light [directional]
My_Green_Spot_Light [spot]

Black highlight indicates active (current) light for editing

Local Lights:

My_Blue_Spot_Light [spot]
My_Yellow_Point_Light [point]
My_Pink_Dir_Light [directional]
My_Orange_Spot_Light [spot]

Local lights are associated with a particular object or group of objects in the scene

Type your name

Light Editor

☒ Headlight On ☒ Show Icon

☒ Light On

Show All Hide All

Color

Direction

Ambient 0 1 0.73

Beam width 1 90 60.33

Cut off angle 20.78

Show/Hide all light icons in the scene

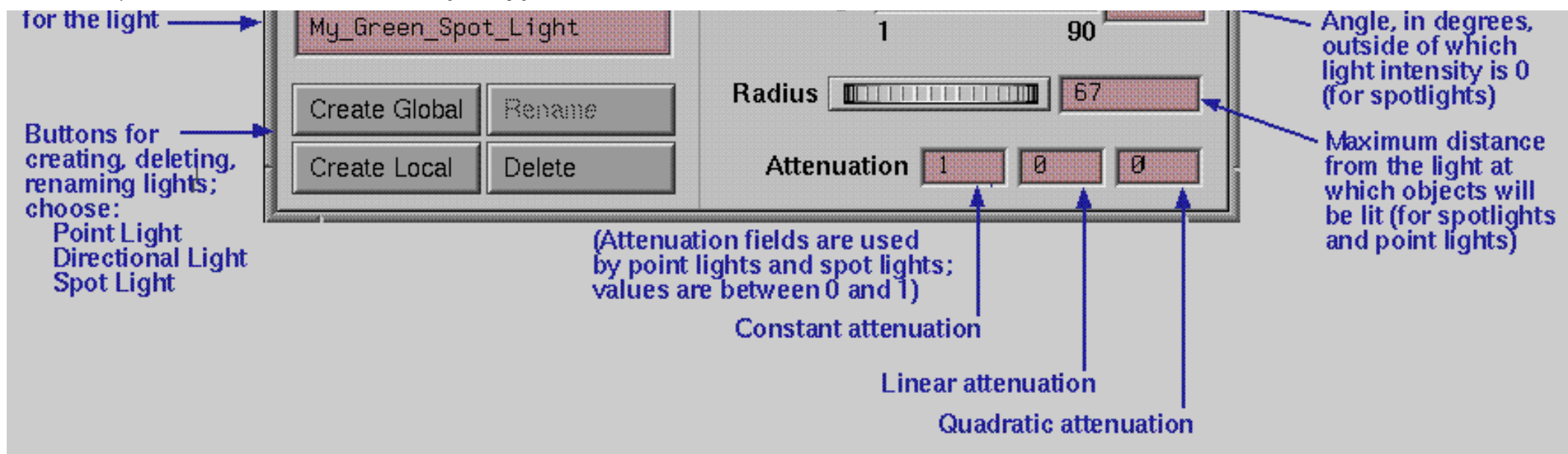
Light color

Light direction (for spot lights and directional lights)

Light intensity

Ambient intensity (between 0.00 and 1.00)

Angle, in degrees, of the main cone of a spotlight



Creating and Editing Lights

on this page: [steps](#) | [local lights](#) | [icons](#) | [headlight](#) | [performance](#) | [scope](#)

You can create three kinds of lights with Cosmo Worlds: *point* lights, *directional* lights, and *spot* lights. Lights have various attributes, including color, location, direction (for directional and spot lights), intensity, and other type-specific attributes such as the beam width for a spot light. Use the Keyframe Animator to animate lights--the headlights on a moving car or a spot light that tracks an object, for instance.

Steps for Creating a Light

1. Click the *Create Global* button to create a global light (or *Create Local* to create a local one).
2. Click the button in the dialog box for the type of light you want to create: Point Light, Directional Light, Spot Light.
3. Type a name for your light in the text box (optional but useful).
4. Use the color wheel to select a color for the light. Left-drag on the small black box to the desired color.
5. Use the slider bar to adjust the intensity of the light.
6. For directional and spot lights, left-drag the blue arrow in the Direction window to change the direction of the light. (You can also manipulate the icon in the main window to change the light's direction.)

Note that if you have a number of lights in the scene, the pink *name* field in the chooser box indicates the active light for editing.

Local Lights

Local lights must be placed under a grouping node in the scene hierarchy. If the *Create Local* button is grayed out, you need to select the parent group for the object or create a parent group if one doesn't currently exist.

Click the "parent" up-arrow button to select the parent group, if there is one. If the *Create Local* button is still grayed out, choose *Edit > Add Parent Group* to create the parent group and enable the *Create Local* button.

Icons

Each light type has a different icon that appears in the scene at the light's location. The icon assumes the color of the light it represents. You can select a light icon and use the manipulator to position and orient the light. (The [Quick Reference](#) card shows the three light icons.)

Headlight

When a new scene is loaded, the headlight is turned on by default if the scene does not contain any lights. If a scene has lights, the headlight is turned off when the scene is loaded so that the effects of the lights are more visible.

A Word about Performance

Lights consume a considerable amount of rendering power, so they should be used in limited quantities in a given scene. A good rule of thumb is to include only two or three lights in a scene for satisfactory performance on most systems. Directional lights are scoped by their parent grouping node, so they scale well with large worlds.

Scope

Point lights and spot lights are always global in scope: they affect everything in the scene, no matter where they are in the hierarchy. Directional lights affect only the objects within their Transform grouping (to see the Transform containing a directional light, use the Outline Editor).

That's the general theory behind lighting. The effects of point lights and spot lights are limited by their radius, so they probably won't actually light up everything in the scene. (Some browsers may not implement this feature). In addition, on some browsers, directional lights are global as well.

Jump to:

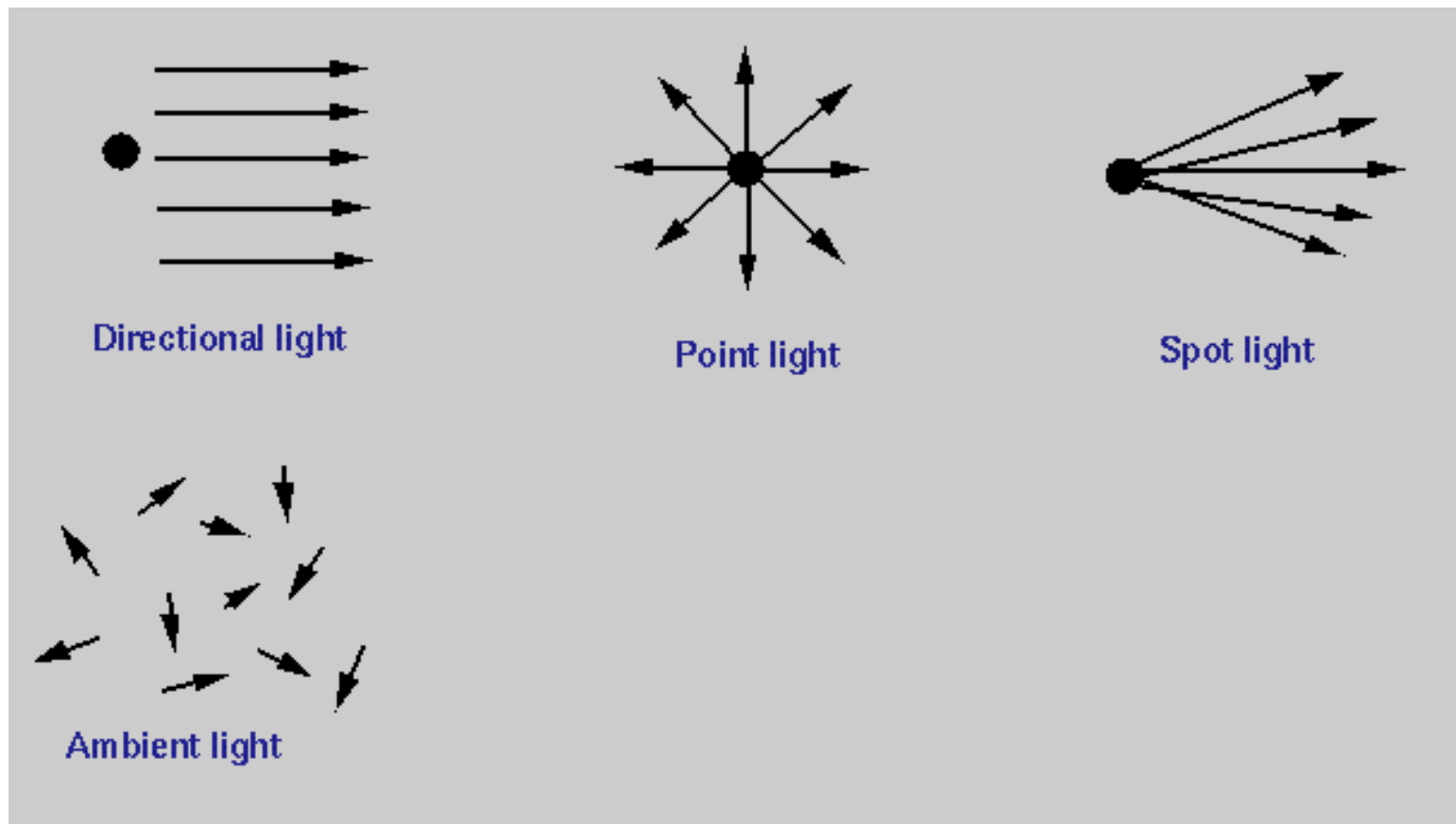
- [Light Editor Quick Reference](#)
- [Tips for Efficient Lighting](#)

Light Types: What's the Difference?

on this page: [ambient lighting](#) | [light attenuation](#)

You can create three different types of lights in a scene:

- Directional lights
- Point lights
- Spot lights



Directional lights are the simplest type of light. They have a direction (but no location). **Point lights**, as the diagram shows, radiate light equally in all directions from a given location. [Spot lights](#) illuminate light along a primary direction from a given location. The spot lights in your scene are analogous to theatrical spot lights, which produce a cone of light that diverges from the light's location.

Each of these lights has a *color*, an *intensity*, and an *ambient intensity*. Point lights and spot lights also have a *radius*, which indicates how far an object can be from the light and still be lit.

Ambient Lighting

Ambient light is the illumination that results from the scattering and reflection of light emitted by all the lights in the scene. It does not have a particular location or direction associated with it. For each light you create, you can specify its *ambient intensity*, which is the intensity of the ambient emission from the light. This intensity can range from 0.00 (no contribution to ambient lighting) to 1.00 (maximum contribution for

this light to ambient lighting in the scene).

If a light is on, its contribution to the scene's overall ambient lighting is computed (for each of red, green, and blue values) by multiplying the light's intensity by its *ambient intensity*, and multiplying the result by the light's value for that color component.

Light Attenuation

If you really want to fine tune the attributes of a point light or spot light, you can use the three *attenuation* fields to specify exactly how illumination falls off with distance from the light. In order, these fields represent three types of attenuation:

- constant - no attenuation; light is constant
- linear - illumination falls off linearly from the light source
- quadratic - illumination falls off based on the square of the distance from the light source

The attenuation is defined mathematically as follows:

$$1/(\textit{attenuation}[0] + \textit{attenuation}[1]*r + \textit{attenuation}[2]*r^2)$$

where r is the distance of the light to the surface being illuminated, *attenuation* [0] is the amount of constant attenuation, *attenuation* [1] is the amount of linear attenuation, and *attenuation* [2] is the amount of quadratic attenuation. The default is no attenuation.

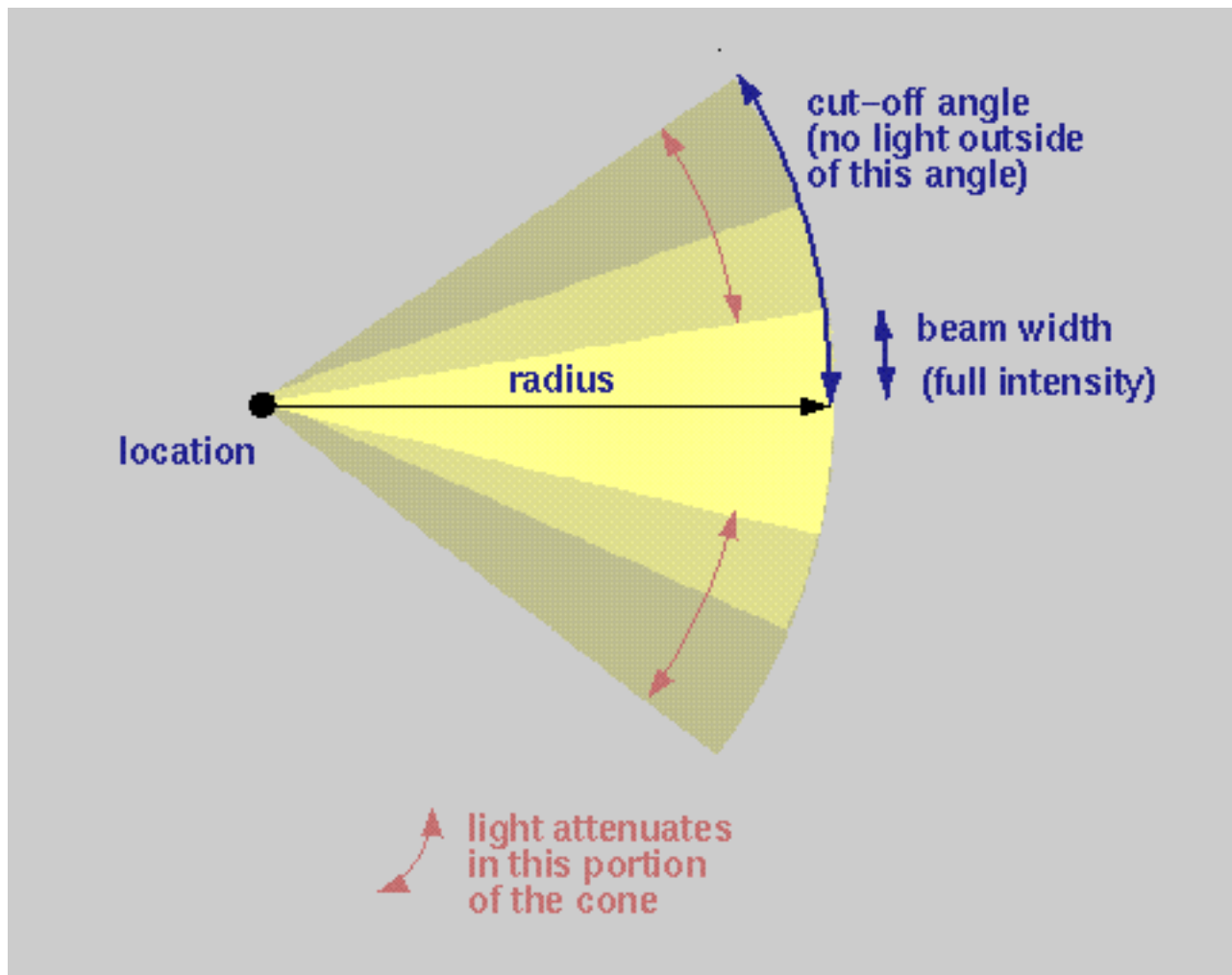
Jump to:

- [More about Spot Lights](#)
- [Using Lights: Try It!](#)

More about Spot Lights

Spot lights allow you to achieve special lighting effects in your scene. Although they consume quite a bit of processing power, you may feel the end result is worth the cost in some cases.

The Light Editor provides fields for specifying the *radius*, *cut-off angle*, and *beam width* of a spotlight, as shown in this diagram:



A spot light produces a cone of light. Anything outside the cut-off angle is not illuminated by the spot light. You can think of the cone of light as an inner cone (specified by *beam width*) and an outer cone (the parts of the one between the inner cone and the cut-off angle). The inner cone of light is full intensity. The outer cone of light becomes gradually fainter as you move from the inner cone boundary to the cut-off angle. The term used to describe how light becomes fainter with distance from its source is *attenuation*.

For the best spot light effect, try a *beam width* of 10 degrees and a *cut-off angle* of 45 degrees instead of the default values.

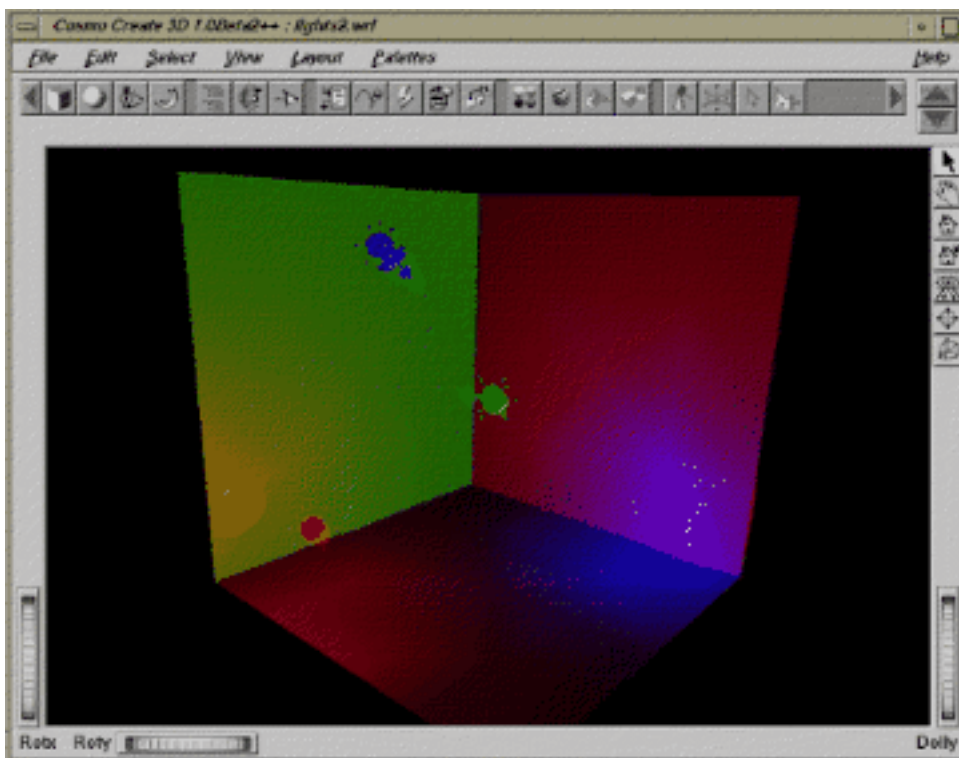
Jump to:

- [Light Editor Quick Reference](#)
- [Using Lights: Try It!](#)

Try It! Experimenting with Lighting Effects

You may be wondering which type of light to use: spot light, directional light, or point light? Probably the best way to learn about different light types is to experiment with them.

1. To get started, in Cosmo Worlds, open the file `/usr/share/Insight/library/SGI_bookshelves/Help/books/CosmoWorlds_UG/Models/lights.wrl`. This file contains three lights: a red point light, a blue spot light, and a green directional light.
2. Open the Light Editor and press the *Show All* button to display the three light icons.
3. Open the Viewpoint Editor and jump to *viewpoint1*, to show the following view:



4. In the Light Editor, select the individual lights and turn them on and off to see their effects. Try moving them around, and use the direction joystick to adjust the direction of the directional light and the spot light.
5. Try creating some new lights. Then adjust their direction, color, and radius (if applicable).

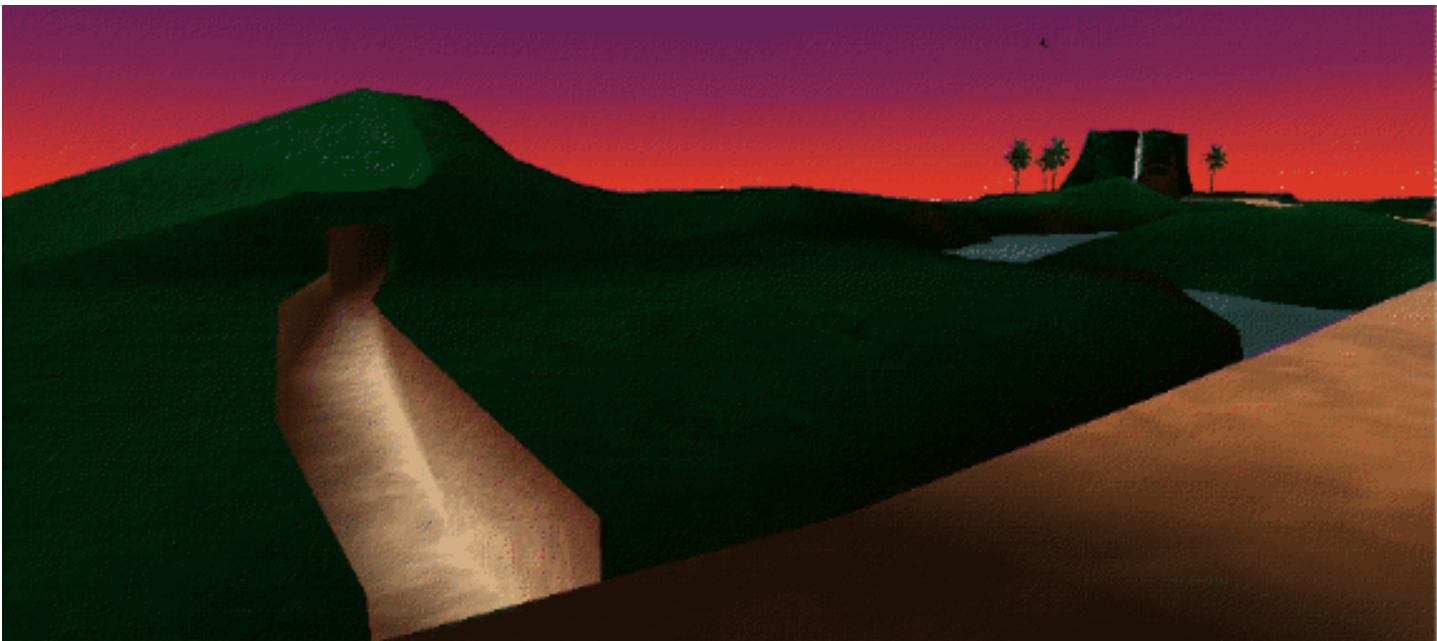
Jump to:

- [Light Types: What's the Difference?](#)
- [Light Editor Quick Reference](#)

Tips for Efficient Lighting

Lights are expensive in terms of performance, so they should be used sparingly. In general, two or three lights per scene is the limit for acceptable rendering speed. A few other tips:

- Spotlights require the most processing power, point lights require somewhat less processing power, and directional lights require the least. Note too that a shape must be finely tessellated in order for you to see the effects of a spotlight (because lighting is done per vertex, not per pixel). These extra polygons also slow down rendering.
- Make sure the materials in your scene have a reasonable amount of emissive color so that objects will still be visible even if lighting is low.
- One way to simulate sophisticated lighting effects such as spotlights is to use per-vertex colors on an object (one designer referred to this as "burning the colors into the shape"). Because the colors are precalculated, rendering speed is improved. The scene shown here uses this technique in place of spotlights:



Jump to:

- [Creating and Editing Lights](#)
- [Light Editor Quick Reference](#)

Adding Navigation Information

on this page: [creating](#) | [know your avatar](#) | [apply and revert](#) | [headlight](#) | [viewer type](#) | [viewer speed](#) | [visibility limit](#) | [advanced](#)

The Navigation Info Editor allows you to describe certain physical characteristics of the viewer and the viewing model. You can specify the default viewer to be used with your scene (Walk, Examiner, Fly, or none). You can also specify certain attributes for an *avatar*, which is a physical manifestation of the user at the current viewpoint in your world. You can specify the size of the avatar, which affects when collisions occur, as well as the height of the tallest objects over which the avatar can step.



Find it: Click the Navigation Info Editor button on the *Editors* palette:

Creating a NavigationInfo Node

Use the *Create Global* and *Create Local* buttons to create NavigationInfo nodes. As with lights and sounds, NavigationInfo nodes can be either *global* (they are placed at the root of the scene hierarchy and are not associated with any object) or *local* (they are placed under a Transform node along with their associated object or group of objects). (See also [Global vs. Local](#).)

NavigationInfo nodes are a type of *bindable* node. The browser binds to the first NavigationInfo node it finds in the file. All subsequent NavigationInfo nodes are ignored unless the file also contains a Script node that binds the browser to other NavigationInfo nodes under certain conditions.

Getting to Know Your Avatar

An *avatar* is a physical manifestation of the user in the scene at the current viewpoint. The *Size* field specifies the allowable distance between the user's position in the scene and an object in the scene before a collision is detected. For example, if you specify a size of 2 and collision detection is enabled, any time the viewpoint comes closer than 2 meters to an object, a collision occurs. The default size is .25 meters.

The *Height* field indicates the distance above the terrain at which the browser should maintain the viewpoint. For example, if you want users to feel about 1.9 meters tall, specify a height of 1.9 so that their viewpoint (their avatar's head) is 1.9 meters above ground level. This field is used only for viewer types that use terrain following, such as the [Walk](#) viewer. The default height is 1.6 meters.

The *Step* field indicates the height of the tallest object over which the viewpoint can step without colliding into it. (You don't want users colliding with every pebble in their path; you simply want the browser to step over the pebbles.) This field is useful for building staircases with dimensions that can

be ascended by all browsers. The default step is .75 meters.

Apply and Revert Buttons

When you click the *Apply* button, the values you've entered are saved as part of a `NavigationInfo` node in the file. When you click *Revert*, the field values return to the previous set of *applied* values. (*Revert* is not exactly like *Undo* because it goes back to the previously applied set of values, not simply the previous value.)

Headlight

Use the *Headlight* check box to specify whether the [headlight](#) should be on or off when the scene is loaded into the browser.

Viewer Type

The *Viewer Type* field allows you to specify which type of viewer the browser should use when the scene is initially loaded.

Viewer Type	Defines this viewer in Cosmo Player*	Viewer used for	Viewer can be changed in browser
Any	Walk (default)	Letting people choose a viewer from within their browsers	Yes
Walk	Walk	Navigating a world by walking through it	No
Examine	Examiner	Looking at a world as if examining a single object	No
Fly	Fly	Navigating a world by floating or flying through it	No
None	No viewer available	Disabling viewer controls	No
Any/Walk	Walk	Navigating a world primarily by walking through it	Yes
Any/Examine	Examiner	Looking at a world primarily as if examining a single object	Yes
Any/Fly	Fly	Navigating a world primarily by floating or flying through it	Yes
Any/None	No viewer available at first	Letting people navigate with keyboard shortcuts	Yes

*Other browsers may refer to the viewers by different names.

Viewer Speed

The *Viewer Speed* field refers to the rate at which the viewer travels through the scene. The default value is 1 meter per second. This value affects panning and dollying in the Examiner Viewer but does not affect rotation speed.

Visibility Limit

The *Visibility Limit* field sets the farthest distance the user is able to see. A value of 0 (the default) indicates infinite visibility. Values for this field must be greater than or equal to 0.

Advanced

See *The VRML 2.0 Handbook*, by Jed Hartman and Josie Wernecke, for a description of how to use a Script node to bind nodes such as the NavigationInfo node to the browser. Chapter 7, "Scripting," describes how the browser maintains a separate stack for each type of bindable node.

Jump to:

- [Controlling Collision Detection](#)
- [Global vs. Local](#)

Global vs. Local

on this page: [selection button](#) | [creating a local object](#)

Certain objects (viewpoints, lights, sounds, textures, scripts, and navigation information) can be created as either *global* objects or *local* objects. What's the difference between the two types?

As far as the objects themselves and how they function in the scene, there is no difference. A global viewpoint has the same effect on the scene as a local viewpoint. A global sound can emit the same sound as a local one. The difference is that a global object is placed in the file at the root (beginning) of the scene description and is not associated with any particular object. A local object is associated with a particular object or group of objects. Use a local light, for example, if you want to associate the light with a particular object or group of objects. If the object is cut and pasted into the scene, the local light follows along with its associated object.

Using the Selection Button

The name panel for local objects lists only the names of the objects associated with the current selection in the scene. For example, if you have a local sound "bang" associated with a box and the box is selected, the name "bang" appears in the name panel for Local Sounds. In complex scenes or scenes authored by someone else, you may have difficulty locating all the local objects. To list all the local objects of a particular type, press the *Select* button in the appropriate editor. You can now select any of the local objects and edit their attributes.

Creating a Local Object

Local objects must be placed under a grouping node in the scene hierarchy. If the *Create Local* button is grayed out, select the parent group for the object or create a parent group if one doesn't currently exist.

Press the *Select Parent* up-arrow button to select the parent group, if there is one. If the *Create Local* button is still grayed out, choose *Edit > Add Parent Group* to create a parent group and enable the *Create Local* button.

Jump to:

- [Light Editor](#)
- [Sound Editor](#)
- [Viewpoint Editor](#)
- [Script Editor](#)
- [Navigation Info Editor](#)

Script Editor

on this page: [Quick Reference](#) | [Scripting Language](#) | [Example](#) |



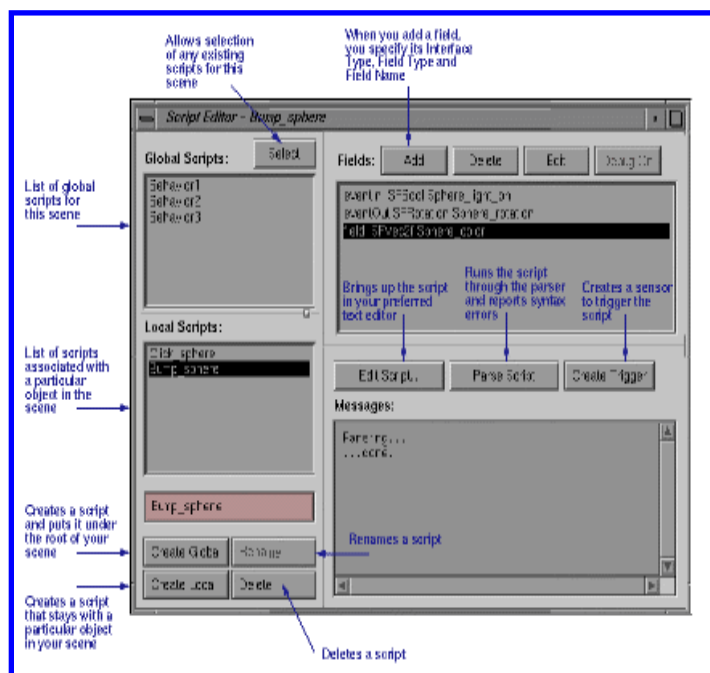
Find it: Click

The Script Editor allows you to define fields and events and create scripts that run within your scene. This editor also includes a button that allows you to create a sensor within the scene to trigger the script.

The script is written in VmlScript, a lightweight, platform-independent programming language that is very similar to JavaScript. VmlScript provides functions that are called when events come into the script, access to fields within the script, logic to operate on the fields, and the ability to send events from the script.

Quick Reference

Here's a quick guide to the Script Editor:



Click image for full view.

Description of the Scripting Language

A detailed description of VmlScript can be found at:

<http://vrm1.sgi.com/moving-worlds/spec/vrmlscript.html>

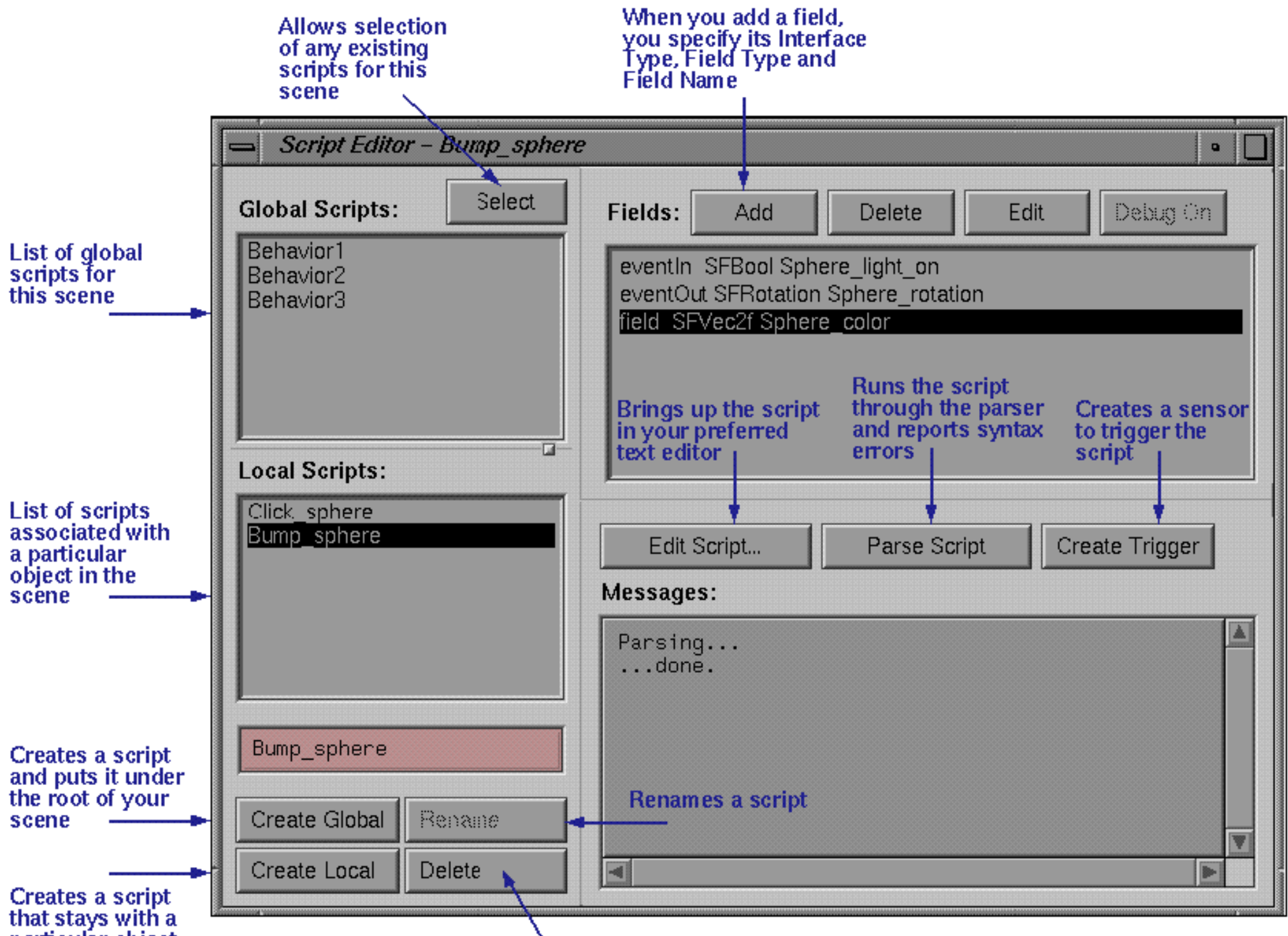
In Cosmo Worlds, the script itself is contained in the *url* field of the Script node, as a single string. The string must begin with *vrmlscript:*. The script defines functions for each eventIn. Internal fields and eventOuts can be used as variables.

Example

Here is a simple example of including a script node in a VRML file:

```
#VRML V2.0 utf8
DEF XFORM Transform {
  children [
    Shape {
      geometry Cone {}
    },
    DEF SENSOR TouchSensor {
    },
    DEF SCRIPT Script {
      eventIn SFBool triggerIn
      eventOut SFFloat fractionOut
      field SFFloat fraction 0
      url "vrmlscript:
        function triggerIn(trigger) {
          if (trigger) {
            fraction = fraction + 0.1;
            fractionOut = fraction;
          }
        }"
    },
    DEF INTERP PositionInterpolator {
      keys [0.0, 1.0]
      values [0 0 0, 3 0 0]
    }
  ]
}
ROUTE SENSOR.isActive TO SCRIPT.triggerIn
ROUTE SCRIPT.fractionOut TO INTERP.set_fraction
ROUTE INTERP.outValue TO XFORM.translation
```

Jump to: [Creating a Script](#)



**particular object
in your scene**

 **Deletes a script**

Creating a Script



Find it: Click the Script Editor button on the *Action* palette:

Use the Script Editor to build special behaviors into your scene file using scripts. A script can be routed to the incoming and outgoing events that are part of the nodes in your scene. It can also be connected to sensors that notify the script when certain things happen--for example, when the user approaches a certain spot, clicks an object, or bumps into a wall.

For information on scripting, see these pages:

- [Scripting Basics](#)
- [Steps for Creating a Script](#)
- [Try It! Case Study for Creating a Script](#)
- [Debugging a Script](#)
- [Tips for Creating Scripts](#)
- [Advanced: Creating a Script From Scratch](#)
- [Script Editor Quick Reference](#)

Scripting Basics

on this page: [script template](#) | [eventIn functions](#) | [outgoing events](#) | [global vs. local scripts](#)



Find it: Click the Script Editor button on the *Action* palette:

You'll need some programming knowledge to use the Script Editor, and you need to understand VRML as well. With this tool, you can create scripts written in VrmIScript, a platform-independent programming language.

Script Template

Use the *Fields: Add* button to specify the basic components of your script, which can be any number of the following:

- eventIn
- eventOut
- field

Each *eventIn* has an associated function with the same name inside the script. When an *eventIn* is sent to the Script node with a new value, this function is called and the new value for the event is passed in.

First, you specify each *eventIn*, *eventOut*, and field with the *Add* button. Field and event names must be unique within the script. When you press *Edit Script*, the Script Editor creates a template file with the events and fields you specified, as well as a template function for each *eventIn*.

You fill in the rest of the template, including the *eventIn* function code and any other necessary code, using your favorite text editor. The Script Editor takes care of adding the Script node that contains your script to the scene file.

eventIn Functions

Every *eventIn* has two predefined arguments: a value and a time. The *value* is the new value of the incoming event. The *time* is a time stamp, of type *SFTime*. When you edit the script, you can change the name of these arguments. If you use only one argument, it is assumed to be the *value*.

Outgoing Events

An *eventOut* is simply a piece of data that the script sends out along a route. It does not have an associated function within the script.

Global vs. Local Scripts

The left panel of the Script Editor allows you to create and name both [global and local scripts](#). Global scripts are scripts located at the root of the scene and are not associated with any object. Local scripts are associated with a particular object or group of objects. If the object is cut and pasted into the scene, the local script follows along with its associated object. To create a local script, first select the object to which the script is to be attached. You can create global scripts at any time, regardless of whether objects in the scene are selected.

Jump to:

- [Scripting Basics](#)
- [Steps for Creating a Script](#)
- [Try It! Case Study for Creating a Script](#)
- [Debugging a Script](#)
- [Tips for Creating Scripts](#)
- [Advanced: Creating a Script From Scratch](#)
- [Script Editor Quick Reference](#)

Steps for Creating a Script



Find it: Click the Script Editor button on the *Action* palette:

These are the general steps required to create a script. The [Try It!](#) page offers a brief case study that creates a sample script and routes to events.

1. Create and name the script. If it is a global script, simply click *Create Global*. If it is a local script, select the object in the scene to attach the script to. Then click the *Create Local* button. (The Script node appears in the Outline Editor as soon as the script is created.) Type a name for the script in the box provided (or use the default name shown).
2. Specify the events and fields used in your script. Choose *Fields > Add*. A dialog box appears, prompting you to select the *Interface Type* (*eventIn*, *eventOut*, or *Field*), as well as the field data type and name. Fields and events cannot have the same names.

(An alternative to using the dialog box is to click the *Edit Script* button and type the fields using the text editor. Advanced users may prefer this technique. See [Advanced: Creating a Script from Scratch](#) for more information.)

3. Create a trigger sensor for the script using the *Create Trigger* button. This step is optional.
4. Now you're ready to program. Click the *Edit Script* button. Your default text editor appears on the screen, with a script template containing the events and fields you specified in step 2. Fill in the rest of the script.

Note: while the text editor window is on the screen, the Script Editor buttons are disabled (except for *Parse Script*). This feature prevents conflicts between the two editor windows in determining which one currently "owns" the script.

5. You can make intermediate saves of the script in your text editor and check the script's syntax by clicking *Parse Script*. Errors print in the *Messages* window.
6. Save the script and close the text editor window. When you save, any new fields added with the text editor appear in the Script Editor *Fields* window.
7. Open the Outline Editor to route the script to the objects you want it to affect.

Jump to:

- [Scripting Basics](#)
- [Try It! Case Study for Creating a Script](#)
- [Debugging a Script](#)

- [Tips for Creating a Script](#)
- [Advanced: Creating a Script From Scratch](#)
- [Script Editor Quick Reference](#)

Try It! Creating a Script



Find it: Click the Script Editor button on the *Action* palette:

Here's an example of how to create a simple script. With this script, when the user clicks the cone in the scene, it doubles in size.

This example uses the [Script Editor](#) to create the script, the Trigger Creator to create a touch sensor, and the [Outline Editor](#) script to the cone.

1. In the main window, create a cone. Select the cone.
2. To create the script, click the *Create Local* button and type a name for the script. This action associates your script with the cone. If the cone is cut and pasted into a new location, your script will follow along with it.
3. Create the trigger sensor that will activate the script. Click the *Create Trigger* button and choose *Touch sensor*. This action adds a new eventIn called *startTime* to the script. It also creates a touch sensor and routes it to the *startTime* event.
4. Add the *outScale* eventOut event to your script using *Field > Add*. (You can also add fields and events directly using the text editor in step 5.)
5. Click the *Edit Script* button. The default text editor appears with a template for your script. ([Click here to view script at this point.](#))
6. Fill in the rest of the code for the *startTime()* function, which scales the object by 2 in all directions. The *outScale* eventOut is of type *SFVec3f*. ([Click here to view finished script.](#))
7. Save the script. Then check its syntax by clicking the *Parse Script* button in the Script Editor. Be sure to save the script before you exit the text editor.
8. Open the Outline Editor. [Create a route](#) from the *outScale* eventOut of the Script node to the *scale* field of the Transform node that contains the cone.

Jump to:

- [Scripting Basics](#)
- [Steps for Creating a Script](#)
- [Debugging a Script](#)
- [Tips for Creating a Script](#)

- [Advanced: Creating a Script From Scratch](#)
- [Script Editor Quick Reference](#)

Outline Editor

on this page: [showing more information](#) | [creating routes](#) |

Find it: Choose *Tools* > *Editors* palette. Then click the *Outline Editor* icon:



The Outline Editor lets you

- browse and edit the VRML source file for your scene
- edit field values
- create routes between events
- make precise selections within the scene file

It shows the entire scene hierarchy, including nodes, fields, field values and types, as well as the routes between events. Node names are in boldface type. Field types and field names are in roman type. Routes are shown to the right of fields and events.

When you select an object in the main window, the Outline Editor highlights the parts of the VRML file that define those objects. Clones (multiple instances) of an object are indicated with the notation [*N*], where *N* indicates how many instances of a node have been created in the file. Edits to one instance affect all instances of the object. If you select an object that has multiple instances, all are highlighted.

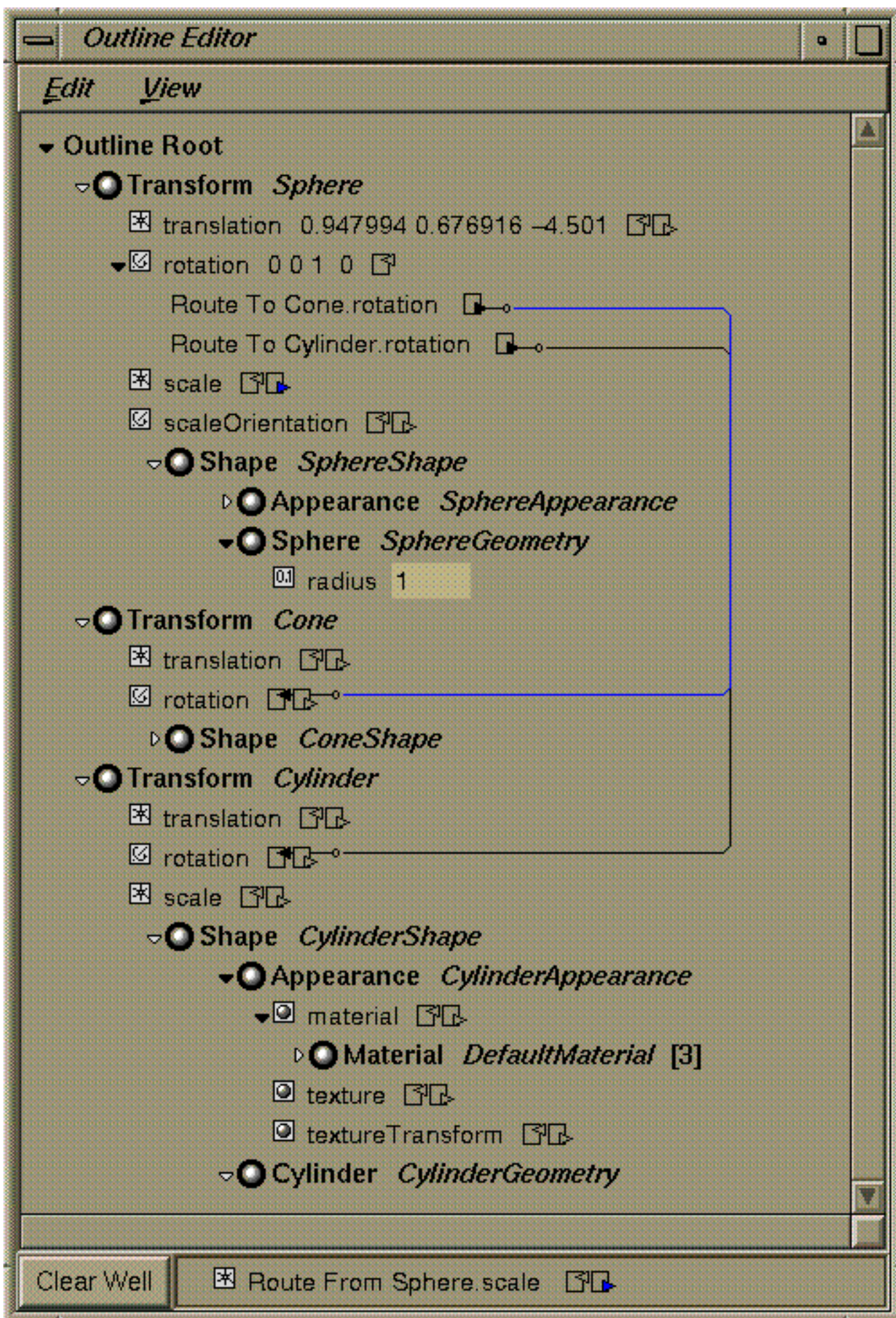
Each node or field has a small arrow to its left. When the arrow on the box points to the right, the information is in its condensed form. When the arrow points down, the information is in its expanded version. Click the arrow next to a node to reveal the fields within that node. Click the field name to reveal the field's values. Each node type has a particular icon that is associated with it.

Creating Routes

Some events can be *routed* to other events. When two events are routed, changing the value of one event automatically changes the values of the events that it is routed to.

Jump to:

- Creating Routes Between Events
- Using the Outline Editor



Creating Routes Between Events

on this page: [a few terms](#) | [creating a route](#) | [removing a route](#) | [rules](#) | [displaying routes](#) | [multiple routes](#)

When you create routes between events, things begin to happen in your scene. You can connect a touch sensor to a lamp to turn on a light, for example. You can also create routes between objects so that when the position, material, or texture of the first object changes, the attributes of the connected object change as well. Routes are created in the [Outline Editor](#).

A Few Terms

Events are of two types: **outgoing** events send values, and **incoming** events receive values. Many fields have both an outgoing event (called an **eventOut**) and an incoming event (called an **eventIn**) associated with them. The Outline Editor uses a graphical shorthand to show these events:



The left-pointing arrow indicates the eventIn, and the right-pointing arrow indicates the eventOut.

When you connect an eventOut to an eventIn (or vice versa), the connection between them is called a **route** (also sometimes referred to as a *wire*).

Creating a Route

Here's an example of how to create a route:

1. Click an outgoing event (the right-pointing triangle).

This step puts the event into the **well**, the small window at the bottom of the Outline Editor. You've now identified the starting point for this route. If you change your mind, you need to click the *Clear Well* button at the bottom of the Outline Editor before you can start a new route.

2. Test out routes. Potential routes highlight in blue when you move the cursor over the pointing triangle associated with the event. If you can't route to a particular event, its triangle is grayed out when you move the cursor over it.
3. When you've decided on a route, click the incoming event (the left-pointing triangle).

This step creates a route between the two events. The event triangles turn black to indicate that they are routed, and the well is cleared.

Tip: If you don't want the well cleared after the route is created, hold down the *Shift* key when you click the mouse in step 3. This action creates the route and leaves the original event in the well so that you can create other routes to that same event.

Removing a Route

To remove a route, hold down the *Ctrl* key and click the mouse over the event (the pointing triangle).

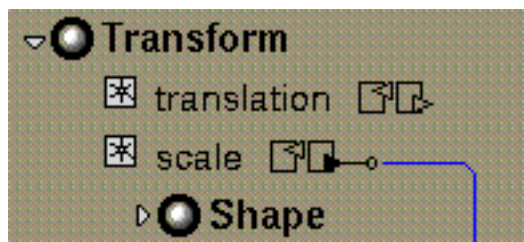
Rules for Routing

The Outline Editor does not let you make "illegal" routes. Here's a list of rules for routing events:

- Incoming events are always routed to outgoing events, and vice versa. You can't route two incoming events to each other, or two outgoing events to each other.
- The field types of the events must be identical. An SFColor event can't be routed to an MFColor event, for instance.
- You can't route an outgoing event to an incoming event for the same field.
- Any SFNode event can be routed to a script, prototype, or debugging node. The Outline Editor has no way of knowing exactly what you're trying to do, so it's up to you to ensure that these connections work. When routing from one SFNode event to another, the Outline Editor checks to be sure that the events have the same name. For example, you can create routes between the *point* fields of two Coordinate nodes, but you can't route a *point* field in one Coordinate node to an *appearance* field in another Shape node.

Displaying Route Information

If you have a lot of routes in your file, displaying all of them at one time may feel a bit tangled. You can click the tail of a route (the small circle on the line attached to the event) to toggle the display of the blue route between events.

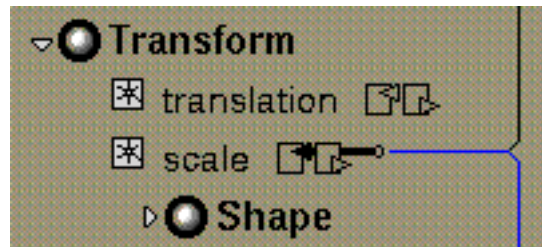


To follow a route to its other end, hold down the *Alt* key and click the tail.

To view detailed information about a route, *Shift*-click the tail of the route.

Multiple Routes

An outgoing event can be connected to multiple incoming events (this is sometimes referred to as "fan out"). Similarly, an incoming event can be connected to multiple outgoing events (this is referred to as "fan in"). In the Outline Editor, both fan in and fan out are represented as thick black lines for the tail of the route. Here's an example of fan-in:



If you *Alt*-click a fan, the routes composing the fan are displayed individually so that you can choose which one to follow.

Jump to: [Outline Editor Quick Reference](#)

Using the Outline Editor

on this page: [selecting objects](#) | [naming objects](#) | [editing fields](#) | [multiple windows](#)

Find it: Choose *Tools > Editors* palette. Then click the Outline Editor button:



To use the Outline Editor, you'll need to understand the basics of the VRML file format. The complete VRML specification is found at <http://vrml.sgi.com>.

Also see *The VRML 2.0 Handbook*, by Jed Hartman and Josie Wernecke, published by Addison-Wesley.

Selecting Objects

When you select an object in the main window, the Outline Editor shows the selection, and vice versa. The Outline Editor uses a white highlight for the master (primary) selection and a gray highlight for secondary selections.

The *View > Follow Primary Selection* option is off by default. If you turn this option on, the Outline Editor automatically scrolls the outline text and opens the node hierarchy to the level necessary to view the current selection.

In some cases, clicking on a node selects an object higher up in the node hierarchy. For example, clicking on a Material node selects the object that contains the Material. Selecting a Shape node that is contained in a Transform node selects the whole Transform, not just the Shape.

Naming Objects

Another important use of the Outline Editor is to assign names to objects in the scene file. The master selection has a text box where you can enter a name for the node. Although it is not necessary to provide names for nodes that are instanced multiple times within a file, it is often useful to provide your own user-friendly names for each use. See *The VRML 2.0 Handbook* for examples of naming and reusing objects.

Use *Edit > Name Primary Selection* to open up the scene file to the level of the current selection in the main window so that you can name the object.

Editing Fields

To edit field values, click a field to reveal its values on the right. Click in the text box to make it active. A blinking cursor indicates that the text box is active. There is only one active text box at a

time, regardless of whether it is used for field editing or naming.

If the text box is not currently active, the primary selection always has a naming box open so that you can easily assign a name to the selection. However, if you've already explicitly opened the text box somewhere else and clicked inside it to make it active, the primary selection will not have this naming box open.

For single-value fields (beginning with SF), click the field to reveal its values. Click again on the field to hide the values.

For multiple-value fields (beginning with MF), click the field to reveal its values. Click a value to edit it. Use the keyboard as follows to move around within an MF field and edit it:

- *Tab* - advance one value
- *Shift-Tab* - go back one value
- *Esc* - cancel
- *Enter* - accept

You can use the *Undo* and *Redo* commands when you're editing field values. For MF fields that have field types with multiple components, more than one value may be highlighted at a time. For example, MFVec3f values are edited in groups of three.

Note: Nodes appearing in italic type in the Outline Editor are for internal use only. You can't edit those nodes or create routes to or from them.

Opening Multiple Windows

You can open multiple windows on the same scene file (use *Outline > New Outline*). When you first call the Outline Editor in a given Cosmo Worlds session, it tries to open as many windows as were open in the layout that you saved, preserving their previous sizes and locations.

Jump to:

- [Creating Routes Between Events](#)
- [Cloned Nodes in Scripts and Prototypes](#)
- [Try It! Case Study for Creating a Script](#)
- [Outline Editor Quick Reference](#)

Cloned Nodes in Scripts and Prototypes

In most cases, you can view the contents of a node by clicking on the small arrow to its left. If a node is instantiated multiple times within a file, the Outline Editor opens only the instance you're currently clicking.

Under certain conditions, however, you'll be able to view and edit only the first instance of the node. This condition occurs when a Script or prototype contains a node that is a clone (instance) of one of its parent nodes. Here's an example:

```
#VRML V2.0 utf8 CosmoWorlds V1.0
```

```
Group {
  children [
    DEF mySphere Transform {
      children Script {
        field SFNode parentNode USE mySphere
      }
    },
  ]
}
```

In this example, the Transform node named "mySphere" contains a Script node with an instance of the "mySphere" node. You can view and edit only the contents of the first instance of this node--in other words, the "parent" instance. You don't need to open the child instance because the parent instance is already open. These child instances have an asterisk appended to their name (or their bracketed number) to indicate that they are special nodes that can't be opened.

Jump to:

- [Creating Routes Between Events](#)
- [Try It! Case Study for Creating a Script](#)
- [Outline Editor Quick Reference](#)

////////////////////////////////// FIELDS //////////////////////////////////

eventIn SFTime startTime

eventOut SFVec3f outScale

////////////////////////////////// BEHAVIOR //////////////////////////////////

function startTime(value, time)

{ }

Jump to: [Try It! Creating a Script](#)


```
////////////////////////////////// FIELDS //////////////////////////////////
```

```
eventIn SFTime startTime
```

```
eventOut SFVec3f outScale
```

```
////////////////////////////////// BEHAVIOR //////////////////////////////////
```

```
function startTime(value, time)
```

```
{  
    outScale[0] = 2.0;  
    outScale[1] = 2.0;  
    outScale[2] = 2.0;  
}
```

Jump to: [Try It! Creating a Script](#)

Advanced: Creating a Script From Scratch

This page offers a few guidelines on how to enter a script directly using a text editor. For a complete description of VRML syntax, see the VRML 2.0 Specification at <http://vrmf.sgi.com>.

Field and Event Declarations

Fields and events are entered after the FIELDS line and before the BEHAVIOR line in the script template. The syntax is

```
interface_type    field_type    name
```

For example:

```
eventIn  SFBool      lightOn
eventOut SFRotation  myRotat
field    SFColor      myColor    1 1 1
```

Fields must have an initial value specified, as shown in this example. Syntax for fields is the same as the file format syntax described in the VRML 2.0 Specification.

Functions

Following the BEHAVIOR line, add a function for each *eventIn* (if you want the event to do something). The form of the function is

```
function eventIn_name( value, time )
{
    // body of function //
}
```

The function can have one or two arguments. If it has only one value, it is assumed to be the *value*.

These functions can call other functions, which have the general syntax:

```
function function_name( arg1, arg2, ... )
{
    // body of function //
}
```

initialize() function

The **initialize()** function is called once when the world is loaded and before any events are processed. This function allows you to set up field values and *eventOut* values. This function takes no parameters. Events generated from it are given the time stamp of when the world containing the Script node was loaded.

shutdown() function

The **shutdown()** function is called when the world is unloaded. It allows you to deallocate memory and perform any other necessary clean-up. This function takes no parameters. Events generated from it are given the time stamp of when the world containing the Script node was unloaded.

eventsProcessed() function

You can define an **eventsProcessed()** function that will be called after some set of events has been received. Some browsers call this function after the return from each *eventIn* function, while others call it only after processing a number of *eventIn* functions. This function takes no parameters. Events generated from it are given the time stamp of the last event processed.

[Click here to view a sample script.](#)

Jump to:

- [Scripting Basics](#)
- [Steps for Creating a Script](#)
- [Tips for Creating a Script](#)
- [Debugging a Script](#)
- [Script Editor Quick Reference](#)

Tips for Creating a Script

A common error is to assign an *eventIn* to an *eventOut* directly by name (see below). It looks correct, but it doesn't work because an *eventIn* in a script is actually a function (with the same name as the event). This function contains data elements (the *value* and *time* arguments). The *eventOut*, on the other hand, is itself a data element.

In the following example, *value* is the data element of the *startFloat* function. This variable is assigned to the *outFloat* eventOut.

```

////////////////////////////////////// FIELDS ////////////////////////////////////////

eventIn  SFFloat  startFloat
eventOut SFFloat  outFloat


////////////////////////////////////// BEHAVIOR ////////////////////////////////////////

function startFloat(value, time)
{
    outFloat = startFloat;    // WRONG!!!
    outFloat = value;         // Correct.
}

```

Jump to:

- [Scripting Basics](#)
- [Steps for Creating a Script](#)
- [Debugging a Script](#)
- [Advanced: Creating a Script From Scratch](#)
- [Script Editor Quick Reference](#)

Debugging a Script

Use the *Debug On/Debug Off* toggle button in the Script Editor to control debugging. The debugging feature allows you to track the current value of any *eventIn* or *eventOut* in the script. When the scene is displayed in Cosmo Player, a debugging window appears each time a value is sent from an event that is routed to the Debugger. The debugging window displays the current value of the event.

To turn on debugging for an event, select the event in the *Fields:* window. Then toggle the *Debug On/Debug Off* button to select *Debug On*. When debugging is on, the Script Editor creates a route from the specified event to the Debugger node. (You can open the Outline Editor to view these routes.) A separate Debugger node is created for each event you specify.

When debugging is turned on for an event, Cosmo Worlds adds a special Debugger node to the scene to allow you to inspect script-related values. The Debugger nodes are removed when you package and publish the scene, since they aren't standard VRML nodes.

The Gallery ([usr/share/data/vrml](#)) has a file containing a Debugger node (*CoRouteDebugger.wrl*). Using the Outline Editor, you can wire this node directly into your scene to check values.

Jump to:

- [Scripting Basics](#)
- [Steps for Creating a Script](#)
- [Tips for Creating a Script](#)
- [Advanced: Creating a Script From Scratch](#)
- [Script Editor Quick Reference](#)

Controlling Collision Detection

on this page: [turning collision off](#) | [turning collision on](#) | [proxies](#)

By default, all objects in the scene are collidable. You can't walk through walls or tables, and you can't end up inside of a planet or other closed solid object. You can use the Collision Editor to turn off collision detection for certain groups of objects (or for the whole scene if you want the user to feel like a ghost who can walk through objects and walls).

You can also use the Collision Editor to specify that a simpler geometry, called a *proxy*, be used in place of the actual object when Cosmo Worlds is testing for collisions. The proxy can be either a box, a sphere, or some other object defined in the VRML file. Using proxies speeds up performance because the test for collisions is much simpler than testing against more complex geometries.

Find it: Click the Collision Editor button on the *Action* palette:



Turning Collision Off

To improve performance, turn off collision detection when it's not needed. Follow these steps:

1. In the main window, select the objects that are not collidable.
2. Press the *Create Collision Grouping* button. This creates a collision group and makes the selected objects children of that group. (As with any group, if the objects are currently children in some other group, they are removed from that group before they are inserted into this group.)
3. Click the *Enable Collision Detection* box to turn off collision detection (the red check in the box disappears).

Turning Collision On

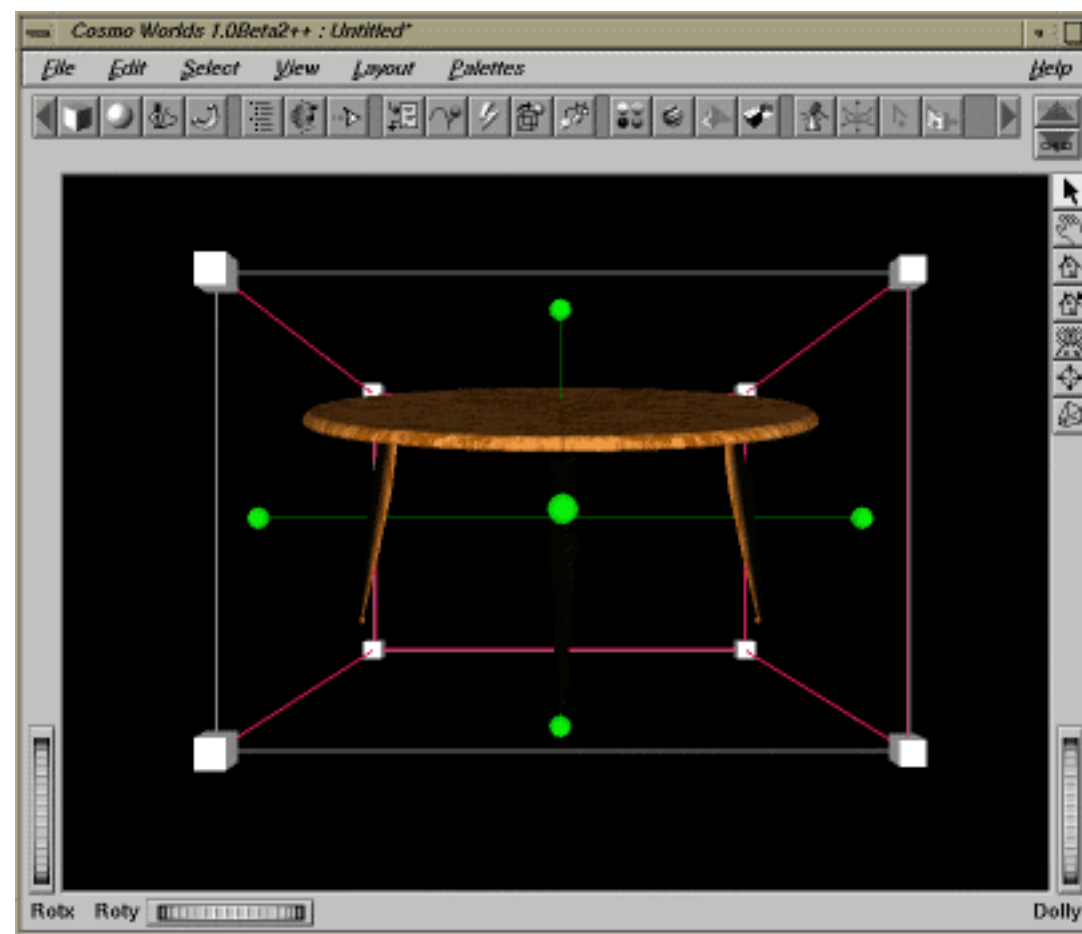
To turn collision back on after you've turned it off, follow these steps:

1. In the main window, select the collision group.
2. In the Collision Editor, press the *Remove Collision Grouping* button. This action ungroups the children, and they now become top-level objects.

Proxies

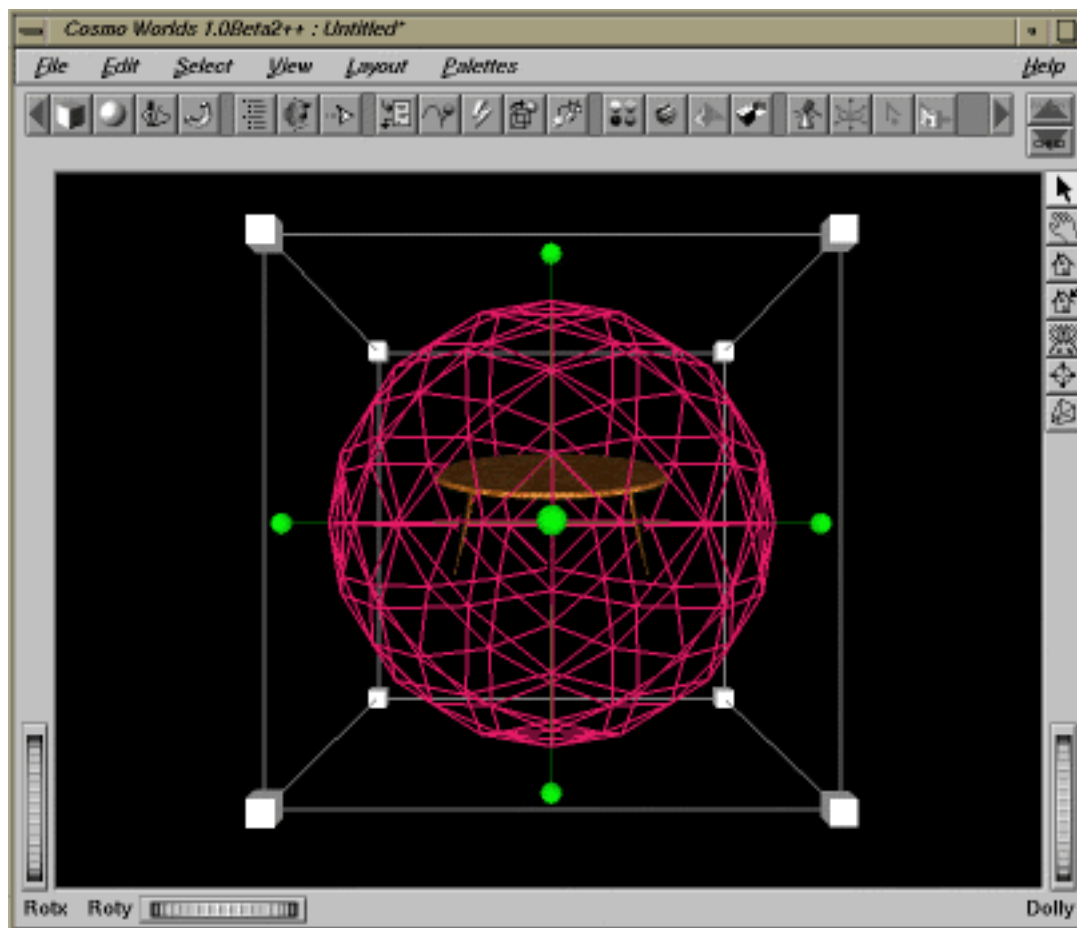
A *proxy* is an object, usually with simple geometry, that is used in place of actual geometries for purposes of detecting collisions. By default, the actual objects are used for collision detection, and no proxies are used.

To use a bounding box (shown below) as a collision detection proxy for a table, create a collision grouping for the table and select it. Then select *Bounding Box* from the *Collision Detection Proxy* pulldown menu.



To use a bounding sphere (shown below) as a collision detection proxy for a table, create a collision grouping for the table and select it. Then select *Bounding Sphere* from the *Collision Detection Proxy* pulldown menu. A bounding sphere is a sphere that completely encloses the collision group.

The *Other* proxy setting is used for proxy objects that are defined directly in the VRML file.



At first, you won't be able to see the collision box or sphere. Use the pulldown menu under *Show Collision Group As* to view the proxy (or both the proxy and the objects themselves). The two images on this page use *Both* for this setting, which displays both the proxy and the actual objects.

Jump to:



- [Navigation Info Editor](#)

Viewer Menu Preferences

The Examiner Viewer and the Walk Viewer both have a Preference pop-up panel. The panels contain almost the same settings; the Walk Viewer alone has a setting for viewer speed and the Examiner Viewer alone has a setting for spin animation and a setting for point of rotation axes.

Find it: [Viewer Menu](#) > *Preferences*

See it:

- [Examiner Viewer preference panel shown with default settings](#) 
- [Walk Viewer preference panel shown with default settings](#) 

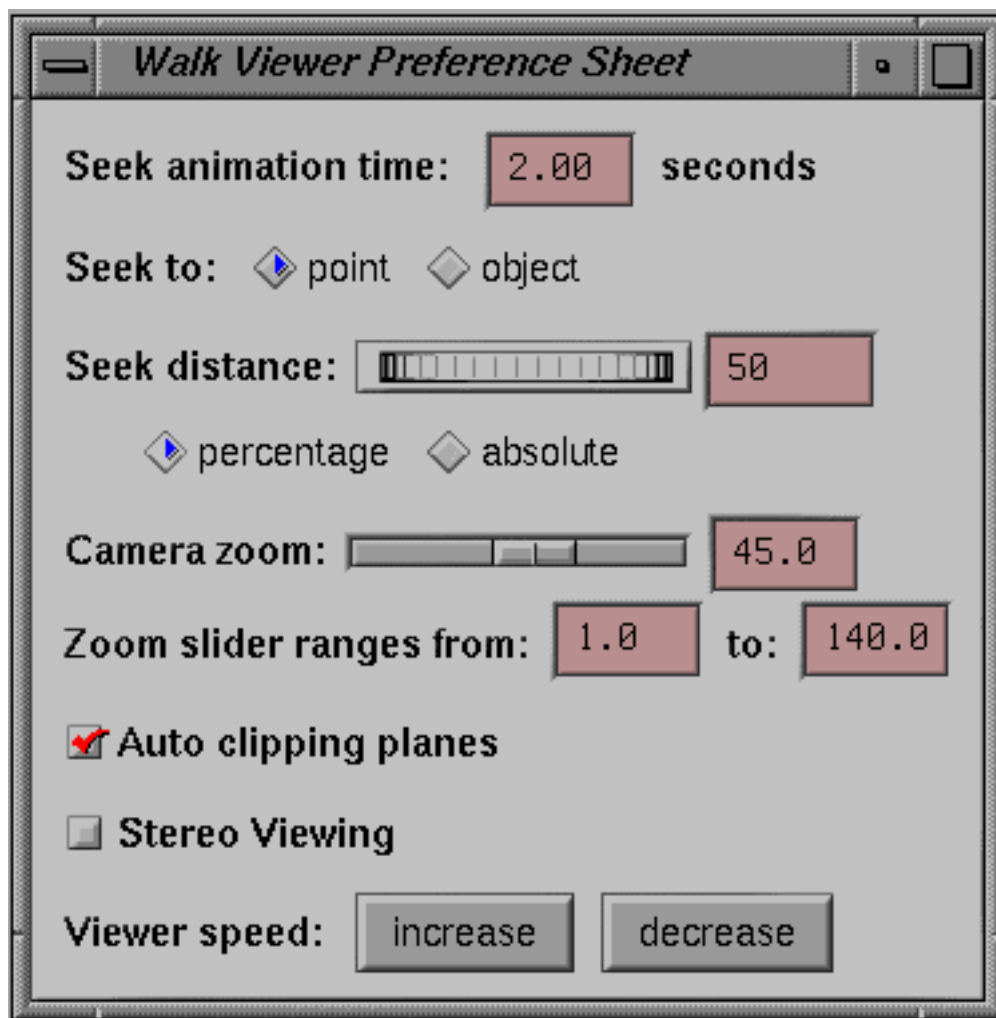
How it works:

- *Seek animation time*--sets the time for the [seek function](#). Set this to 0 for instant seek.
- *Seek to*--allows two levels of accuracy for the [seek function](#). *Seek to point* uses the selected point to dolly the camera to. *Seek to object* uses the center of the object to dolly the camera to.
- *Seek distance*--defines how far the camera moves; measured in meters or the percentage of distance between the camera and the seek destination point.
- *Camera zoom*--changes the angle of the camera lens. Produces a wide angle lens distortion when you zoom out and a flattening of the model or scene when you zoom in. The camera does not move when you zoom; only the lens angle changes.
- *Zoom slider ranges from*--the default setting for the zoom slider range is 1.0 to 140.0. You can adjust the minimum and maximum ranges to as low as 0 and as high as 180 degrees.
- *Auto clipping planes*--default is on, which means Cosmo Worlds automatically sets the [clipping planes](#) for the best dynamic fit for the object on the screen. If you turn *Auto clipping planes* off, you can move the clipping planes by using the rotation thumbwheels that appear.
- *Stereo viewing*--turn stereo viewing on to change the camera rotation using the thumbwheel that appears. Use this option if people looking at your scene have special stereo glasses and hardware. Default is off.
- *Enable spin animation (Examiner Viewer only, view mode)*--turns on or off the ability to quickly rotate and let go of an object to make it spin. To spin, click and drag an object with the hand icon. Release the mouse button while still dragging the mouse, and the object continues spinning. To stop the spinning, click anywhere in the scene. Default is on.
- *Show point of rotation axes (Examiner Viewer only, view mode)*--shows the point about which

the camera rotates. When you turn it on, you can also change the size of the axes using the thumbwheel that appears. Default is off.

- *Viewer speed increase/decrease (Walk Viewer only)*--increases or decreases the viewing speed in meters per second. This is useful when navigating through worlds that are not at the right speed. Note that these settings do not get saved with the VRML file. Authors should use [Navigation Info](#) to save the viewer speed in the file.





Selecting and Grouping Objects


on this page: [select all or part](#) | [select multiple objects](#) | [group](#) | [add to existing group](#)

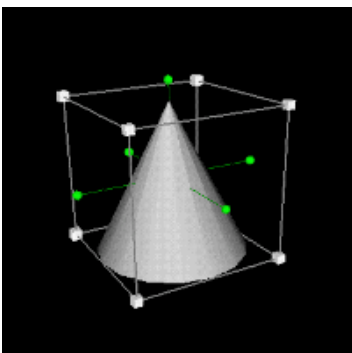
Group objects to specify several objects as a whole. You can use grouping to organize parts of a model--to make it easier to move and edit several objects at a time, for example. Or you can use grouping to define more advanced hierarchies--to make it easy to articulate parts of the model during an animation, for example.

The basics of selecting and grouping:

- Click an object to select it; *Shift*-click to extend the selection to include additional objects.
- Use the grouping functions from the *Edit* menu to group, ungroup, add to the existing group, and create a new parent group. The grouping functions are explained in more detail below.
- Use [group hierarchies](#) to specify spatial relationships between objects in a group. For example, a robot leg that articulates at the knee and foot joints should be grouped so that moving the upper leg also moves the knee, lower leg, and foot. (This is where *Create Parent Group* and *Add to Existing Group* become useful.)


To select all or part:

- Use pick (arrow button  found on [Viewing toolbar](#)) to choose an object. When you click, the object's manipulator appears. This is a bounding box around the selected object with knobs for translating, rotating, and scaling the object:



- Choose *Edit > Select All or Deselect All* to quickly select and deselect all the objects in your scene. **Shortcuts:** *Ctrl-/* and *Ctrl-*
- Clicking on the background deselects all objects.
- *Shift*-click a selected object to remove it from the selection.



- **Within a group:** Use the *Parent* and *Child* arrow icons (found to the right of the top toolbar) to select parts of a grouped object. Select *Parent* or *Child* displays the group hierarchy up to the part you have currently selected. For example, if you select the shin of a robot leg that is part of a group hierarchy *Thigh, Knee, Shin, Foot*, then pressing the *Child* button will only select as far as the shin.
- You can also use the Outline Editor to select parts of an object by clicking the object's Transform. [The Transform becomes highlighted once you click it.](#) 

To select multiple objects:

Select an object and use *Shift*-click to select additional objects. The last object selected in this manner is the **master selection** and has manipulator handles. The other selected objects display bounding boxes with no manipulator handles. However, you can still place the cursor over the individual bounding boxes to move the group. Resizing and rotating is controlled from the master selection. Moving, rotating, and resizing affects all of the objects in the multiple selection at the same time.

When you click on another object, or in the scene, the objects are deselected. Pressing *Shift*-click again removes objects from the selection.

To group objects:

1. *Shift*-click to select the objects you want to group.
2. Choose *Edit > Group*, or use *Ctrl-g*. The objects will now be grouped, with a single manipulator. The grouped objects are treated as a single object whose axis is the center of the group. The objects remain grouped unless you choose the group and choose *Edit > Ungroup*, or use *Shift-Ctrl-g*.

See [Creating Group Hierarchies](#) for more details on grouping.

To add a new object to an existing group:

1. Select the object you want to add to the group.
2. *Shift*-click to select the group, which makes the group the master selection.
3. Choose *Edit > Add to Group*.
4. Use *Edit > Detach From Group* to remove the object that you added to the group.

This technique applies to adding a new object to any existing group, including those defined by the

[LOD](#) and [Switch](#) editors. See [Creating Group Hierarchies](#) to learn how *Add to Group* differs from *Group*.

To create a new parent group:

1. Select the objects above which you'd like to create a new parent group.
2. Choose *Edit > Create Parent Group*.

See [Creating Group Hierarchies](#) to learn when to use *Create Parent Group*.

Creating Group Hierarchies

on this page: [about](#) | [group](#) | [add to group](#) | [create parent group](#)

About Group Hierarchies

A group hierarchy specifies parent and child relationships within the group. The simplest group hierarchy occurs when you select objects and use *Edit > Group*. This groups all the objects at the same level under a new parent, making the objects the children of the new parent. This grouping treats the separate objects as a single object with its own manipulator, so you can move and resize the objects as a whole. To create a multi-level group hierarchy, create multiple groupings using *Edit > Group*. Also use *Edit > Add to Group* and *Edit > Create Parent*. Group, Add to Group, and Create Parent are described in detail below.

Group hierarchies are especially important when setting up animations for objects that are connected by joints; for example, the legs of a robot.

Edit > Group

In an example using a robot leg made up of a foot, shin, and thigh, a simple group looks like this:

Group Foot, Shin, and Thigh



The purple area represents a parent Transform node which holds the children (three Transform nodes containing information for the Foot, Shin, and Thigh).

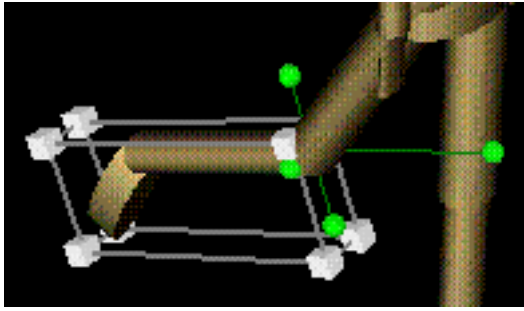
The Outline Editor might display this grouping as:

- Transform (Parent for the following children)
 - Transform (containing Foot)
 - Transform (containing Shin)
 - Transform (containing Thigh)

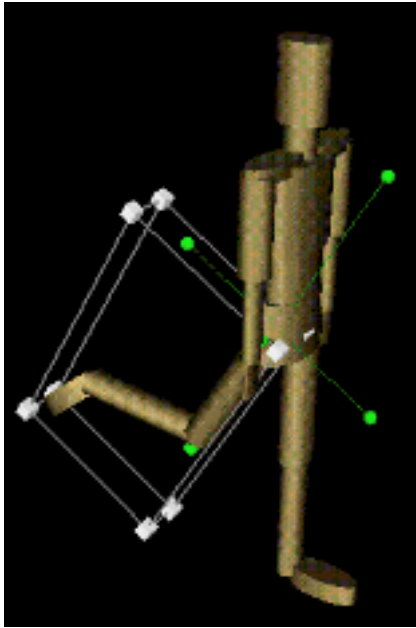
This type of group will allow you to move the foot, shin, and thigh as a single unit.

But you want to do something more complex: Suppose you want to move just the foot, then the

shin with the foot:



And you also want the thigh movement to include the shin and the foot:



You'll need to group the foot and shin together, then add the thigh as a separate group containing both the foot and shin. The steps for this operation follow:

1. Select and group the foot and shin (use *Edit > Group* or *Ctrl-g*)
2. *Shift*-click to add the thigh to the selection and choose *Edit > Group*.

The outcome produces a group hierarchy that looks like this:

Group Foot and Shin; *Group* Thigh



The Outline Editor might display this grouping as:

- Transform (Parent group shown in purple above)
 - Transform (containing Thigh)
 - Transform (Parent group shown in orange above)
 - Transform (containing Shin)
 - Transform (containing Foot)

This type of grouping lets you easily articulate the robot leg.

See [The Keyframe Animator: Try It!](#) to step through an example of animating a similar group hierarchy.

Edit > Add To Group

Suppose that you want to add a knee to the robot leg described in the previous section. For this, you'll add the knee to the existing foot and shin group so that all three of these objects move together.

The order in which you select to add to a group is important. First select the object you want to add to an existing group, then *Shift*-click to select the group.

1. Choose the knee.
2. Choose the grouped foot and shin. Because the thigh, shin, and foot are grouped together, you need to use the *Select Child* button to select just the foot and shin:



3. Choose *Edit > Add to Group*.

The outcome produces a group hierarchy that looks like this:

Add To GroupKnee



The Outline Editor might display this grouping as:

- Transform (Parent group shown in purple above)
 - Transform (containing Thigh)
 - Transform (Parent group shown in orange above)
 - Transform (containing Shin)
 - Transform (containing Foot)

- Transform (containing Knee)

Create Parent Group

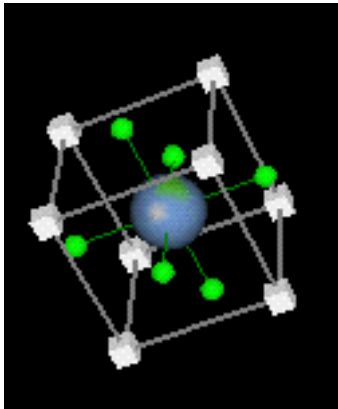
In some cases you'll want to create a parent group. For example, suppose you are creating an animation of the Earth's rotation. The Earth spins around its own axis and it also orbits the Sun. You can create a parent group to specify the Sun's rotation that includes the rotating Earth:

1. Select the Earth.
2. Choose *Edit > Create Parent Group*.

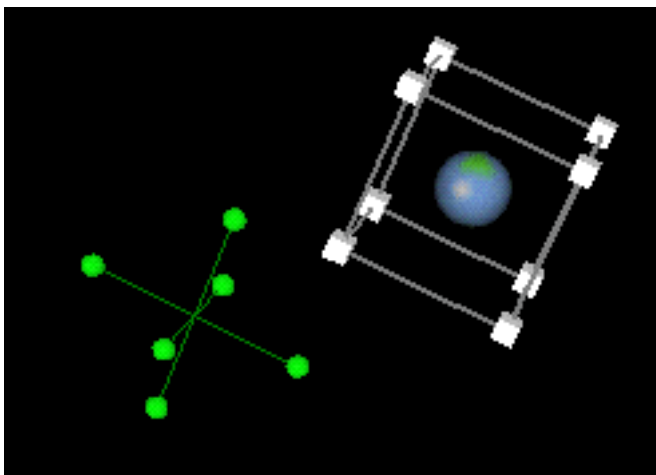
The parent group has its own manipulator with its own center of rotation. You can move this center by pressing *Ctrl* and dragging one of the green balls on the manipulator.



When you create an animation, use the Select Parent and Select Child buttons to toggle between the Earth's local rotation (the child selection) and its rotation around the Sun (the parent selection).



Select Child to set the Earth's local rotation.



Select Parent to set the Earth's rotation around the Sun. (Press Ctrl and drag a green ball to move the center of rotation away from the parent's manipulator.)

Jump to:

- [Selecting and Grouping Objects](#)
- [Animating Models](#)

1. In Cosmo Worlds, open the file */usr/share/Insight/library/SGI_bookshelves/Help/books/CosmoWorlds_UG/Models/anim.wrl*.
2. Start the Keyframe Animator by clicking the bouncing ball button.
3. Select the object you're going to animate. Click anywhere on the object to select the whole object (body, arm, and hand).
4. To create a new animation, choose *Animation > New animation*. To save the animation with the scene, choose *File > Save* from the Cosmo Worlds window.
5. Name the animation. Choose *Animation > Change name*. Type a name into the dialog box (for example, *salute*) and click OK.
6. Add a member to the animation by choosing *Animation > Add member*. The selected object is added as a member of the animation. Once you've added an object as a member, all of the objects that are grouped under it are automatically eligible to be animated. As you select them, you'll see lines appear in red underneath the member line.

Part II: A Few Adjustments

Here are a few adjustments you can make to various settings in the Keyframe Animator.

1. Choose *View > Show time in seconds* to display the time in seconds.
2. Set the length of the animation to 2 seconds. Choose *Animation > Set duration*. Type 2 into the dialog box and click OK. Note that, by default, frames are in tenths of a second.

Part III: The Action Begins

Now, for the action. In this section, you'll set the time for each keyframe, pose the figure, and then record the pose for each keyframe.

1. Move the scrub bar to the end of the animation, either by clicking in the time trough or by dragging the scrub bar.
2. Select the lower arm and hand (by clicking the Child down arrow in the right-hand corner of the Cosmo Create 3D window) and move the arm to its final position.
3. Click the Master record button (the blue box next to the word "Master").

The keyframes at position 0 are recorded (they turn blue). In addition, the keyframes at position 20 are recorded. They are currently red because they're selected. Since these are the first keyframes on these lines, you get an automatic keyframe at frame 0 as well, recording the initial pose to interpolate from. The keyframes at frame 0 are blue because they are not selected.

4. Select the parent of the lower arm/hand group by clicking the Parent up arrow in the right-hand corner of the Cosmo Worlds window. Then move the arm to an intermediate position in its waving gesture.
5. Click the Master record button.
6. Practice moving the scrub bar, selecting portions of the robot and posing them, and recording frames.
7. To delete a keyframe, select it by clicking it (it turns red). Press *Backspace* to delete it.
8. Preview the animation by clicking the Play button.
9. To save the animation, save the file for the scene.

Part IV: Dragging Selected Keyframes

1. Here's another interesting thing you can do. Go to keyframe 20 and click the keyframe for the UpperArm/Xfm property line. The keyframe turns red because you've selected it (and the keyframes of its parents, the Body and Master keyframes, turn red as well).
2. Move the scrub bar to frame 5. Drag the selected keyframes to the new time. click the Master record button. Now the robot moves its upper arm before it moves the lower arm.

Part V: Detaching the Hand and Scaling the Hat

1. If you want to make your animation similar to the one supplied with this tutorial, select the hand (the cone) for the final two frames. First, move the scrub bar to the selected frame. You can select the hand using the Child arrow in the Cosmo Worlds window, or you can select it by highlighting Hand in the Names panel of the Keyframe Animator. Now, position the hand and record the keyframes.
2. Move the scrub bar, scale the cone (now a hat), and click Record.

Congratulations! You've now learned the basics of animating an object in a scene. For more practice, try the tutorials listed below for selecting, copying, and pasting keyframes and for animating viewpoints.

Jump to:

- [Keyframe Animator Reference](#)
- [Selecting, Copying, and Pasting Keyframes](#)
- [Animating a Viewpoint](#)

Animating a Scene

You'll use a variety of tools to animate the objects in your scene:

Keyframe Animator

The Keyframe Animator lets you create sequences of poses, called "keyframes," and interpolate between them to create a smooth animation. You can animate the movement of anything in your scene such as objects, lights, and viewpoints. You can also animate many other attributes such as colors, texture transformations, and the shapes of PEP objects.

Script Editor

The Script Editor facilitates the creation of scripts that drive certain behaviors in your scene and react to user events.

Outline Editor

The Outline Editor lets you create routes between events. Routes are the mechanism for sending values from one node to another.

Switch Editor

The Switch Editor lets you create switch nodes that select which child to use within a particular group. Switches are usually used with scripts that select the child within the switch group.

Trigger Creator

The Trigger Creator lets you add sensors to the scene. These sensors start an animation or execute a script when a particular event occurs.

Viewpoint Editor

The Viewpoint Editor lets you define viewpoints. The Keyframe Animator lets you animate a viewpoint by defining keyframes for its placement.

Nodes and Tools Library

This directory -- `/usr/share/data/vrml`--contains useful nodes and tools you can add to your file.

Creating a Switch

on this page: [creating a switch](#) | [adding objects](#) | [tip](#)

Find it: Click the Switch Editor button on the *Action* palette:



A switch is a group that displays only one of its children at a time. Typically, a switch is used with a script that controls which child is displayed. A switch can be used to create simple animations or to display different versions of an object under different conditions. See Chapter 7, "Scripting," in *The VRML 2.0 Handbook* by Jed Hartman and Josie Wernecke for an example of a script that controls a switch.

Creating a Switch

To create a switch:

1. In the main window, select the objects that will be the children of the switch.
2. Click the *Create Switch Grouping Over All Selections* button.
3. Choose the initial child to display: click either the box labeled *None* or the box with the child number. Children are numbered in order, starting with 0. The order of the children within the switch is the order in which they were selected. To change the selected child, click the up or down arrow, or type a number in the box.

Adding Objects to an Existing Switch

To add objects to an existing switch:


1. In the main window, select the objects to add.
2. Shift-select the switch group last (so that it's the master selection).
3. Choose *Edit > Add to Group*.

Tip: If you want all the objects to be in the same location, use *Selection > Center Snap Target* to move the snapping target to the center of the current master selection. Another useful command in the Layout menu is *Move Selection Center to Snap Target*, which moves the master selection so that its center is at the snap target. Secondary selections move with the master.

Jump to: [Script Editor Quick Reference](#)


▼ Outline Root


▼ ● Transform **ALL MOTION**

✱ scale 


▷ ● Transform *HIPS_DATA*


▼ ● Transform *LT_THIGH_MOTION*

✱ translation 

✱ center 


▼ ● Transform *LT_THIGH_DATA*


✱ translation 

✱ scale 

▷ ● Shape

▼ ● Transform *LT_SHIN_MOTION*

✱ translation 

✱ center 

▷ ● Transform *LT_SHIN_DATA*

▷ ● Transform *LT FOOT MOTION*

▷ ● Transform *RT_THIGH_MOTION*

▷ ● Transform *ABDOMEN_MOTION*

Creating Multiple Levels of Detail

on this page: [steps](#) | [example](#) | [performance LOD](#) | [ranges](#) | [modes](#) | [selecting](#) | [adding a level](#) | [removing a level](#)

A Level of Detail (LOD) grouping contains any number of child objects, referred to as *levels*. For a Distance LOD, you specify the distance at which each level is displayed. For a Performance LOD, you allow the browser to determine which level to display for optimal performance.



Find it: Click the Level of Detail Editor button on the *Action* palette:

Steps for Creating an LOD Grouping

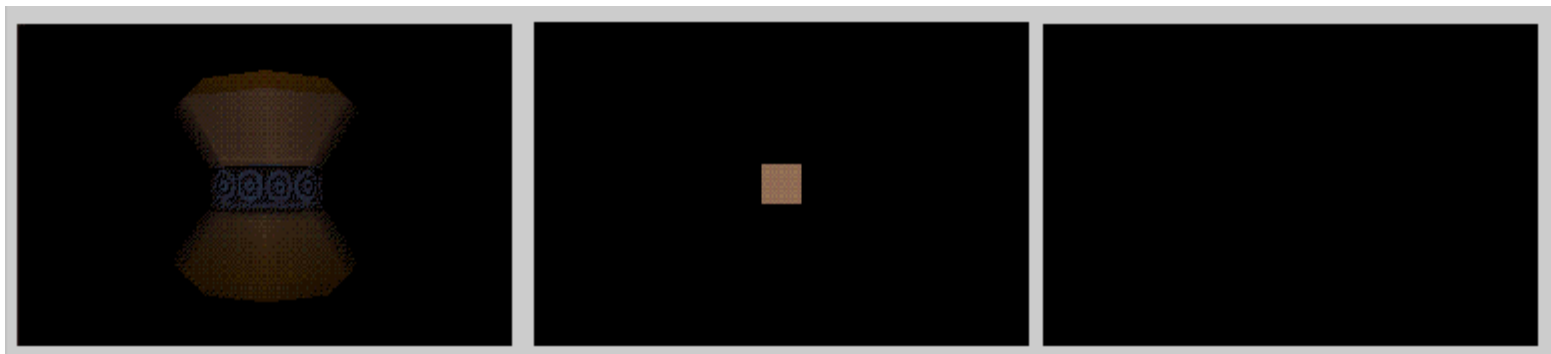
Follow these steps to create a simple LOD grouping:

1. Create a cube, a sphere, and a cone. Select them in that order.
2. Press *Create LOD Grouping Over All Selections*.
3. The first level (the cube; level 0) is now displayed. To cycle through the other levels, click in the range bar to move the blue highlight. When you select the middle range, the sphere (level 1) is displayed. When you select the last range, the cone (level 2) is displayed. You can also use the *Current Level* arrows to cycle through the levels, or type a level into the *Current Level* text field.
4. To see the LOD grouping in action, click the *Active LOD* box. Now dolly in and out of the scene and watch the levels change as the distance changes.
5. Next, try adjusting the ranges so that the levels change at new distances. To adjust a range, click and drag one of the tick marks on the range bar to the left or right. Be sure the *Active LOD* box is checked, and then try dolly in and out of the scene again to see the levels change.
6. When an object is very far away, you often want it to disappear from the scene entirely. To create this effect, press the *Append Empty Level* button. The scene now contains a fourth level (level 3), which displays nothing.
7. To equalize the ranges, deselect and reselect the LOD group in the main window. The ranges adjust to a more even spacing.

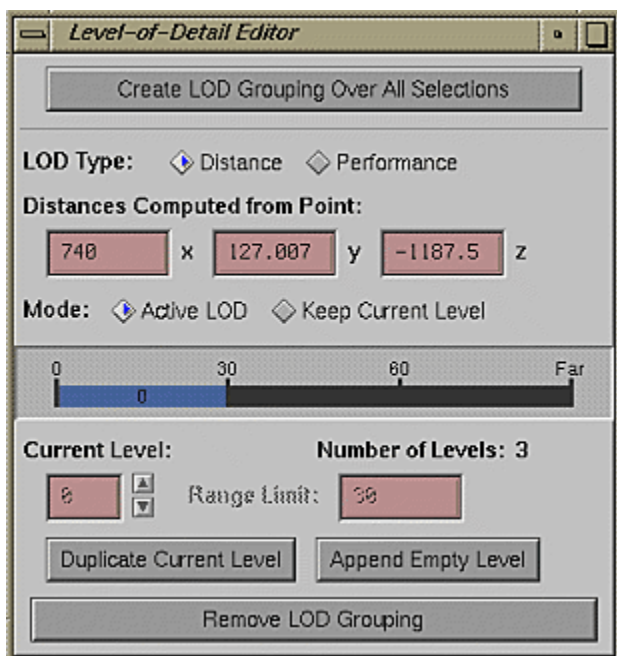
Example

Now that you have a basic feel for how an LOD grouping works, open the file `/usr/share/Insight/library/SGI_bookshelves/Help/books/CosmoWorlds_UG/Models/brazier.wrl`.

This LOD has a textured brazier for the first level, a brown cube for the second level, and an empty third level, as shown below. Its structure is typical of a useful LOD grouping, which usually contains a least one detailed object for close viewing, one stripped-down version, and one empty group for very far distances. You could also create another intermediate-level version of the first object using the Polygon Reduction Editor.



Select the object and open the Level of Detail Editor to examine the settings. Here's what you'll see:



Performance LOD

Click the *Performance* box to convert an LOD to a Performance LOD, which allows the browser to select the appropriate level based on rendering performance, not distance. When you choose the Performance LOD option, the range bar shows "Best" at the left and "Fastest" at the right. The browser selects the level with the lowest index that it can render while still maintaining an acceptable frame rate. Some browsers may ignore this option.

When you've selected the Performance option, the *Distances Computed from Point* and *Range Limit* text boxes are grayed out.

How Ranges Work

For Distance LODs, distances are measured from the user's viewpoint to the center point of the object. You specify this center point by typing values into the *x*, *y*, and *z* text boxes shown under the label *Distances Computed from Point* in the Level of Detail Editor. The default center point is the origin (0 0 0). The brazier example shown earlier has a center of (740 127.007 -1187.5).

The *Range Limit* text field shows the range limit for any range but the last one. It reflects the limit of the currently selected range (in blue on the range bar). When you type a new value into this field and press *Enter*, the new value is shown on the range bar.

Two Modes: Testing and Editing

The LOD Editor has two modes, one for testing LOD groupings (*Active LOD*) and one for editing the levels (*Keep Current Level*). Use the *Active LOD* mode when you want to test the transitions between one level and the next. As you zoom in and out of the scene, the transition between levels should feel gradual; you don't want to see a sudden "pop" as levels switch. Use the *Keep Current Level* option to stay with one level so that you can perform detailed editing on it.

You can click and drag the range bar tick marks in either mode.

Selecting an LOD Grouping

If the LOD grouping is selected, switching between levels doesn't change the current selection. If a level or part of a level is selected, switching between levels *does* change the current selection.

When you save the file, the *Active LOD* mode should be active. Note, though, that if you save, then select the LOD grouping, and then select an individual level, the mode switches automatically to *Keep Current Level*. This change enables you to make changes to the selected level. When you deselect the level and then select the entire LOD grouping, the mode returns to *Active LOD*.

Adding a Level

Adding a level to an LOD grouping is the same as adding a child to any group. See [Selecting and Grouping Objects](#) for details.

Removing a Level from an LOD Grouping

Removing a level from an LOD grouping is the same as removing a child from any group. Select the level and then cut it or detach it from the group.

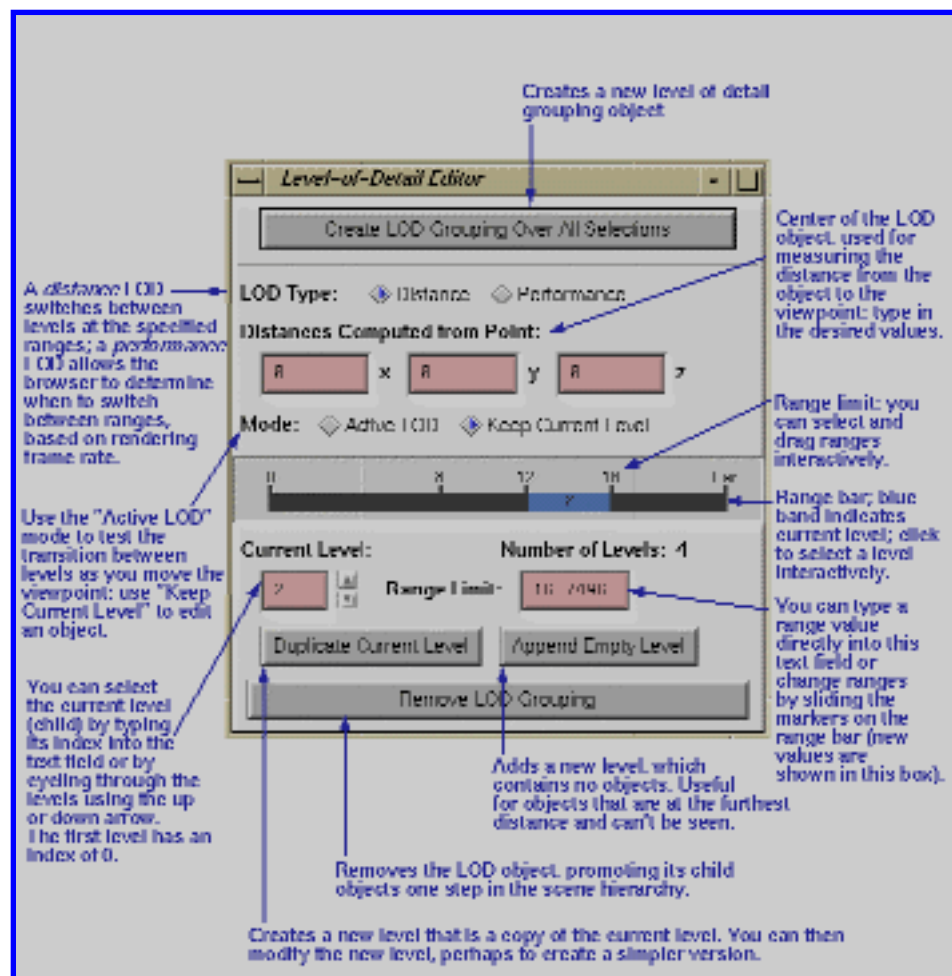
Jump to: [Level of Detail Editor Quick Reference](#)

Level of Detail Editor

Use to: Create alternate representations of an object, with varying levels of detail. Once you've created the alternate representations, you can specify the distances at which to display each representation, or *level*. Or, you can omit the ranges, creating a *performance LOD* (Level of Detail) grouping that allows the browser to select the appropriate level for optimal performance.



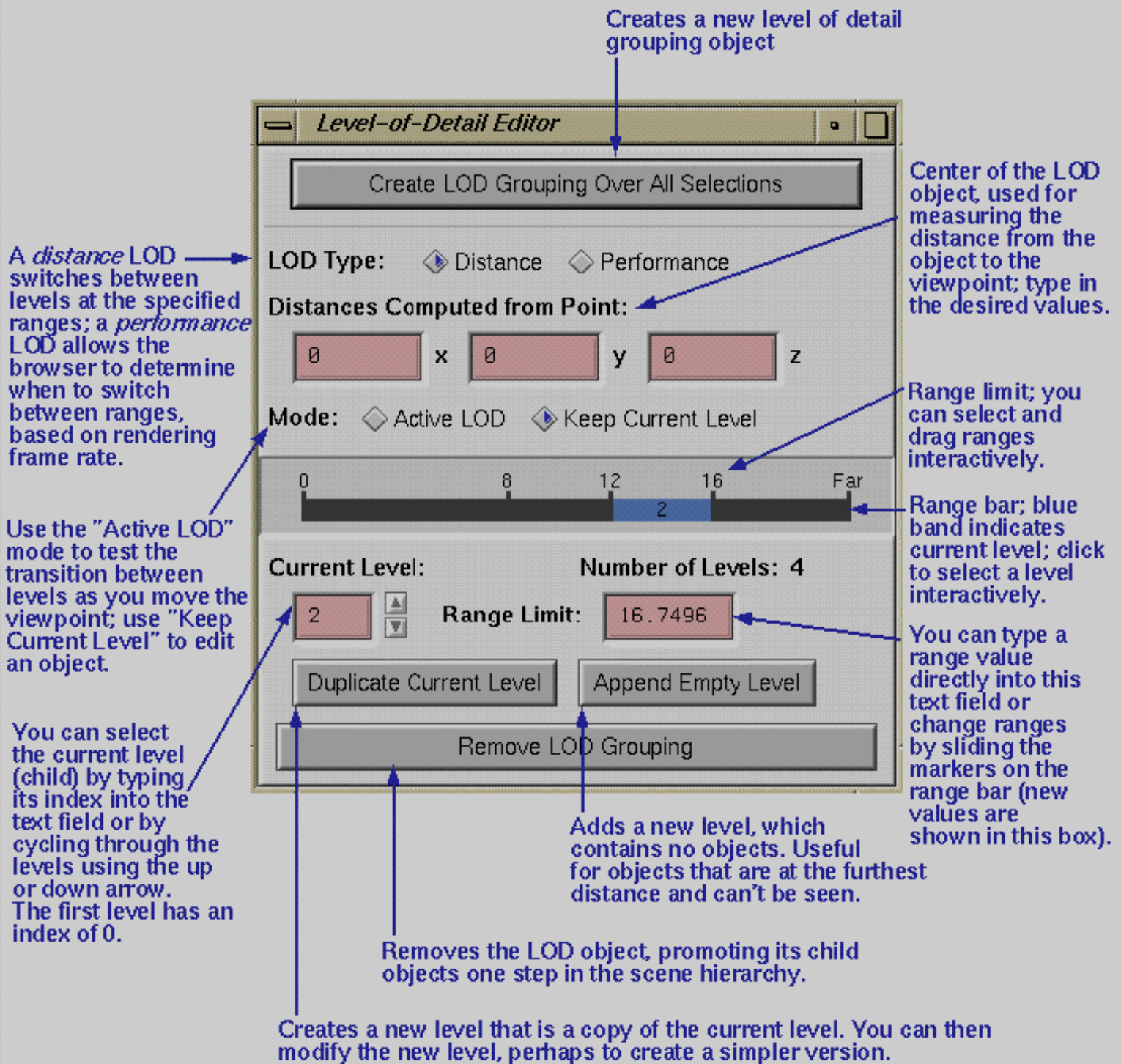
Find it: Click its button on the *Action* palette:



Click image for full view.

Jump to:

- [Creating Multiple Levels of Detail](#)
- [Polygon Reduction Editor](#)



Polygon Reduction Editor

on this page: [how it works](#) | [tips for best results](#) | [tips for creating LODs](#)

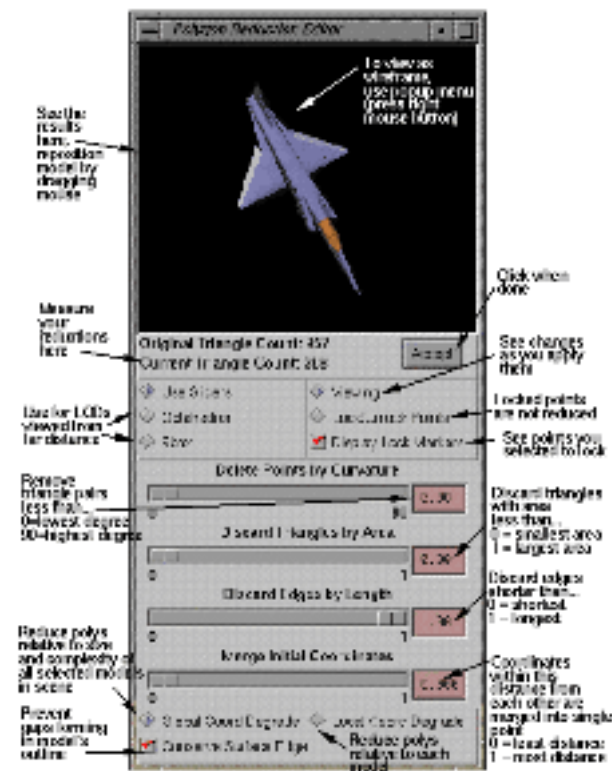
Find it: Click its button on the *Editors* palette:



Use to:

- Reduce polygon count of an object or group of objects in your scene, while keeping the shape's integrity.
- Create multiple [Levels of Detail](#) for an object.

How It Works





[Click image for full view.](#)

Viewing Your Edits

Click *Viewing* to see your reductions in the editor's preview window. This window is an [Examiner Viewer](#) with its controls hidden. Use *Alt-drag* to adjust the object in the preview window.

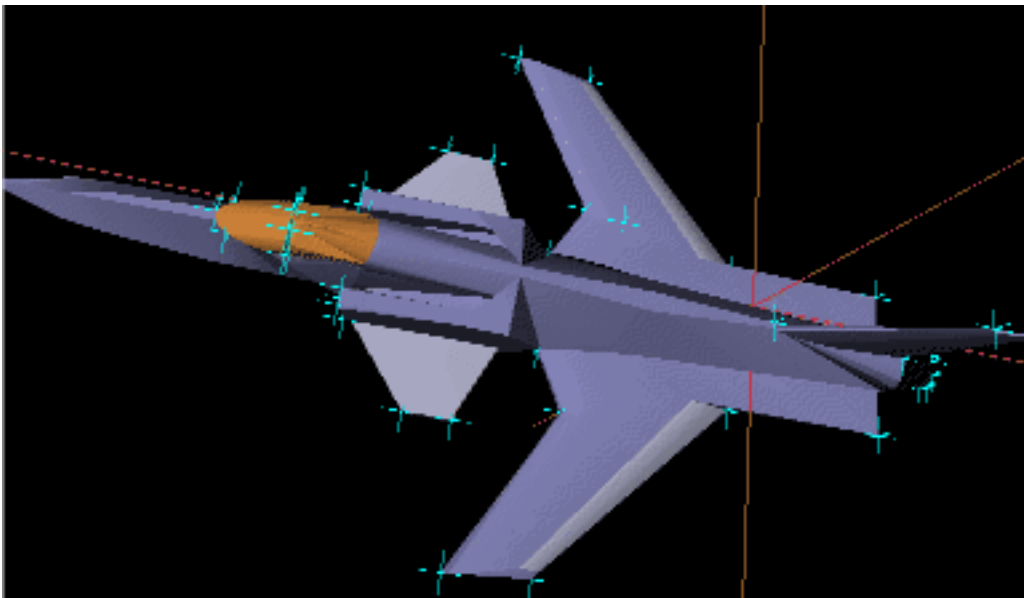
Using the Sliders

Use the sliders to experiment with different types of polygon reduction. Also see [Tips for Best Results](#).

- **Delete Points by Curvature:** Deletes points and re-triangulates the shape to compensate for lost points. Measures the [dihedral angle](#) between each adjacent pair of triangles and removes those triangle pairs that have a common dihedral angle less than the value you specify with slider. 0-90 corresponds to the degree of dihedral angles.
- **Discard Triangles by Area:** Finds the area of triangles in the object and gets rid of the smallest triangles. Move the slider to adjust the area of the triangles that are discarded.
- **Discard Edges by Length:** Calculates the lengths of all edges of the object and discards the smallest. Move the slider to adjust the value of the edges that are discarded.
- **Merge Initial Coordinates:** Merges nearby points into a single point if they are within a certain distance of each other. Move the slider to adjust the distance at which nearby points are merged.

Tips for Best Results

- Play with the different sliders to see which combinations yield the best result. To start, try the *Discard Edges by Length* slider with *Conserve Surface Edge* selected. This pair reduces many polygons while keeping the original shape of the object.
- Try sliding a bar a little bit, clicking *Accept*, and repeating--this incremental reduction can work well with some objects. You can always undo mistakes in the main scene with *Ctrl-z*.
- If you find a slider combination that gives you a pretty good result except for a few points, turn on *Lock/Unlock Points* then click points to specify those that should not disappear when you reset the slider. You can toggle on and off the markers for *Lock/Unlock Points* by clicking *Display Lock Markers*. It is often useful to work in hidden line or wireframe mode while choosing points to lock ([Popup Menu](#) > *Display*).



Polygon reduction with Lock/Unlock Points. Notice the red cross-hairs in the scene. These move with your mouse cursor to help you identify and select the correct points you want to lock. Locked points appear as light-blue Xs.

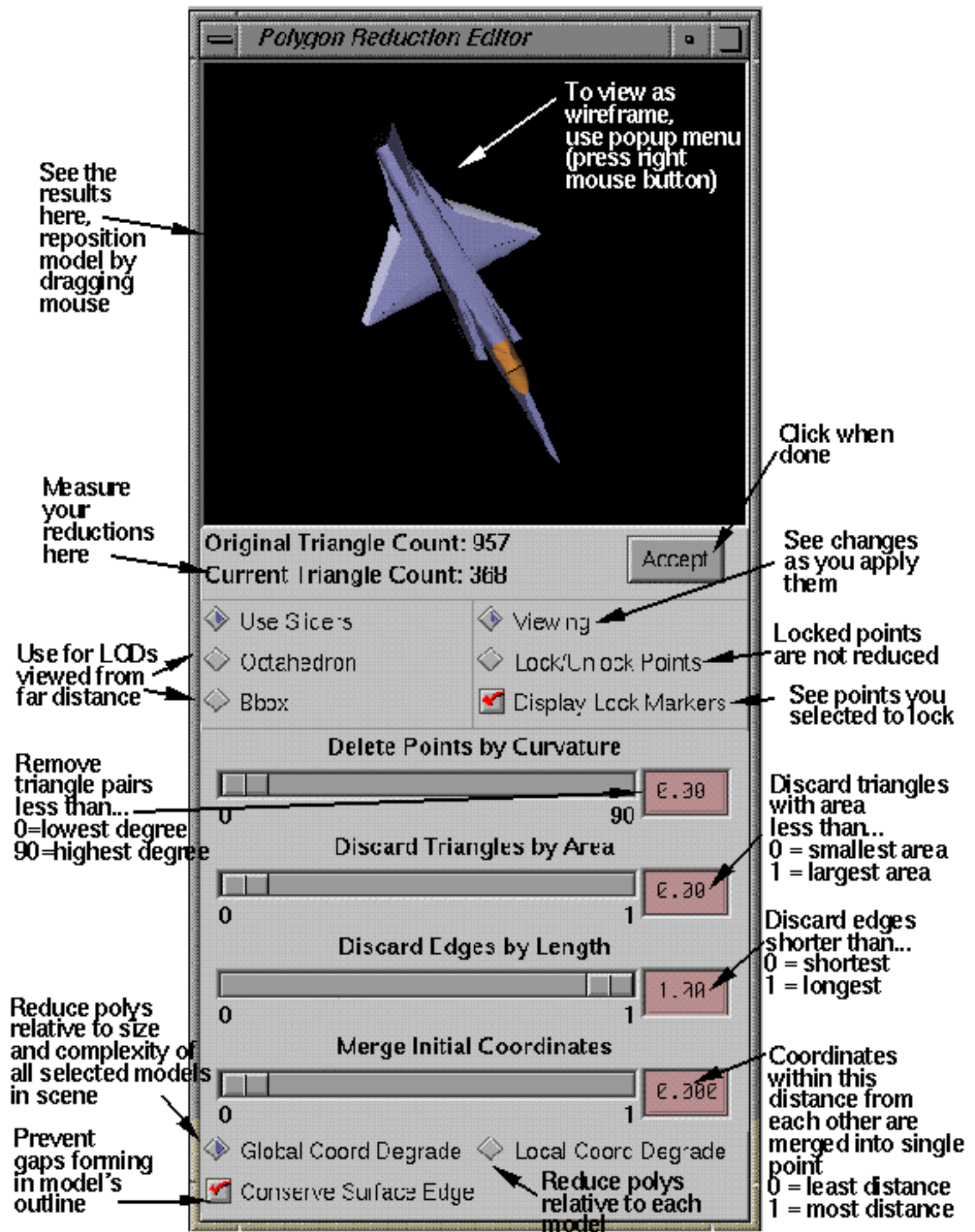
- Instead of *Lock/Unlock Points*, you can first try *Conserve Surface Edges*. This helps keep the object's original shape intact when you reduce it. Sometimes holes may appear; if this happens, using *Lock/Unlock Points* gives you a better result.

Tips for Creating LODs

- Create a new file for each LOD, then specify its use in your scene by using the [Level of Detail Editor](#).
- When working in the Polygon Reduction Editor, periodically adjust your view of the model at the range the LOD object will be viewed. This helps you gauge how much detail you really need.
- Use *Octahedron* or *Bbox* for the lowest level of detail. Use the sliders for LODs that will be viewed more closely.

Jump to:

- [Optimizing Your Scene](#)
- [Reducing Polygons \(Task Summary\)](#)



Optimizing Your Scene

Keep in mind when you create your VRML scene that visitors to your world will be using different browsers on Silicon Graphics, PC, or Macintosh platforms. They may have a T1 connection to the Internet, or they may be connected at 14.4 by modem.

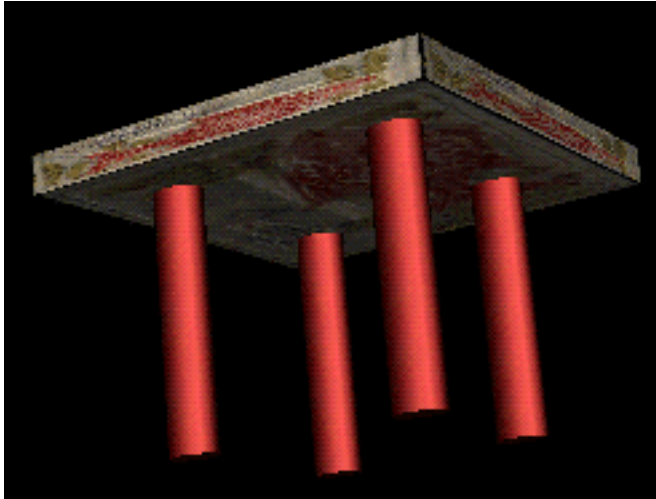
Your goal is to create a scene that most people can quickly navigate through. This means modeling objects with a low polygon count and using some optimizing tricks to keep your scene graph small:

- Model objects separately, and save them as individual files. Create a separate file for your scene, and import objects or inlines as needed.
- [Create clones](#) instead of copies when you use multiple instances of an object in your scene.
- Use the [Primitive Counter](#) to keep track of each object's polygon count.
- Use the [Polygon Reduction Editor](#) to reduce each object's polygon count. Also use this tool in conjunction with the [LOD Editor](#) to create and place versions of an object at lower levels of detail. Place all but the invisible and lowest level of detail versions in the scene as inlined objects using the [Inline Editor](#).
- Rebuild complex models with the [Polygon Copier](#).
- Turn on backface culling in models by using the [Normal Doctor](#).
- Split complex scenes into several smaller scenes, linked through anchors using the [Link Editor](#).
- Use the [Billboard Editor](#) to "paste" 2D images instead of creating a full 3D object.
- Use the [Texture Editor](#) to give an object complexity without increasing its polygon count.
- Create [VRML text](#) instead of geometric text (the latter has a very high polygon count).
- Use the [Collision Editor](#) to specify collision detection only where needed. For example, instead of creating your whole scene as a collision area, select only those objects a visitor will try to walk through or bump into.
- Use sensors to trigger actions needed only in particular parts of your scene. For example, instead of creating permanent lights, you can use the [Switch Editor](#) to create a "hot spot" where a light will be triggered only when needed.
- [Package](#) your final scene to VRML, and test on different platforms with different browsers to make sure your audience can quickly download and navigate your world.

Cloning Objects


on this page: [cloning vs. copying](#) | [creating and pasting](#) | [selecting](#) | [pep modeling](#) | [grouping](#)

Cloning, or instancing, is an [optimizing](#) technique. If you use the same object multiple times in your scene, clone the object instead of copying it because cloning references the original object's geometry instead of creating a whole new object. Clones share the same appearance but can be resized and placed independently. PEP modeling one clone affects all clones sharing the same reference.



Cloned table legs. Note that the legs share the same material but are located in different coordinates.

Cloning vs. Copying

Cloning produces an identical copy of the object that **shares** that object's geometry and appearance information. Copying produces an identical copy of the object that has its own geometry and appearance information. [See the cloning vs. copying diagram.](#) 

Creating and Pasting Clones

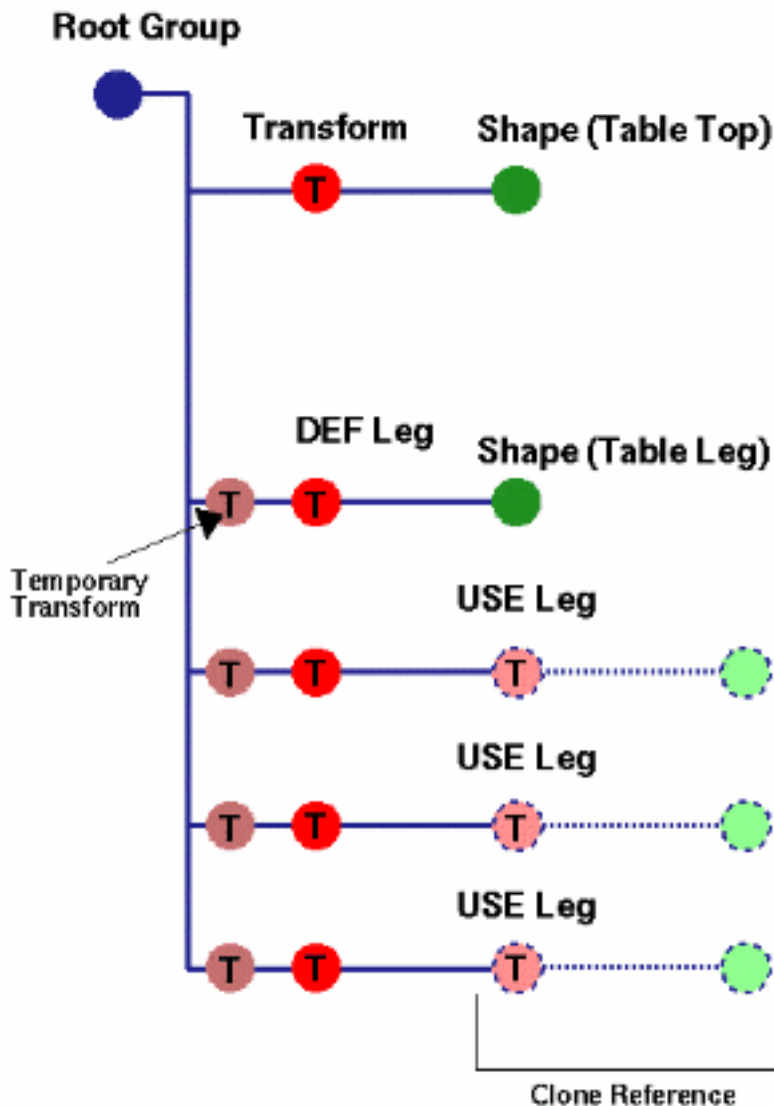
Edit > Clone creates a clone reference of the selected object and places it in the clipboard. *Edit > Paste Clone* pastes that clone into the scene. You can consecutively paste multiple clones into the scene.

Shortcuts:

- Clone = *Shift-Ctrl-c*
- Paste Clone = *Shift-Ctrl-v*

Selecting Clones

To select clones and perform translations and rotations on the correct Transform, you should be familiar with the grouping hierarchy that cloning creates. The diagram below represents the group hierarchy formed for the [table top and its cloned legs \(shown above\)](#).



Grouping hierarchy created by cloning. Note that the dotted "USE Leg" are clone references. They refer to "DEF Leg". (These terms are used in the VRML file.) The solid red Transform above them contains translation and rotation information for each clone.

Key points about selecting clones:

- Creating a clone creates a new group hierarchy.
- The new hierarchy looks different in the Outline Editor and in the VRML file. The Outline Editor shows clone information as Transforms. Each clone has an extra Transform that lets you move, rotate, and resize the clones independent of the original. The VRML file defines the object to which the clones refer with *DEF*; the clones are defined by *USE*. In the diagram above, the table leg is defined as *DEF Leg*, and each clone refers to that leg with *USE Leg*.



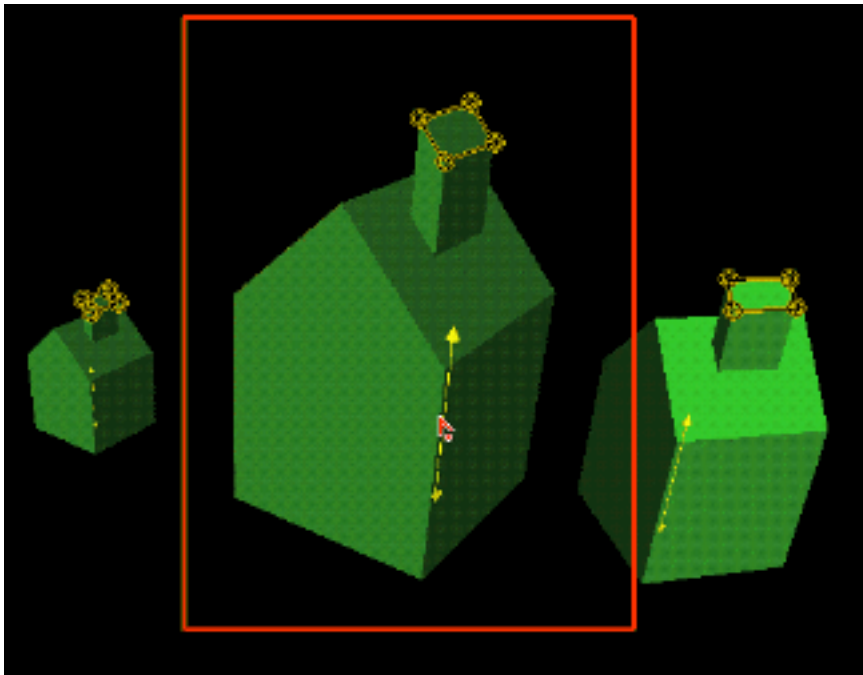
- If you use the Select Parent and Select Child button, notice that there are three levels you can cycle through:
 - Select a clone--that's the parent. Move, rotate, or resize the clone at this level.
 - Click *Child*--that's the temporary Transform shown in the diagram above. If you move, rotate, or resize the clone at this level, an extra Transform is added to the VRML file. This extra Transform always appears in the Outline Editor.
 - Click *Child* again--that's the cloned object. Notice that all the clones for that object appear with manipulators in your scene. You need to be at this level if you want to PEP edit the clones (see [PEP Modeling Clones](#) for details).

PEP Modeling Clones

[PEP modeling](#) one clone affects all the clones sharing the same reference. Before you can enter the PEP editor, choose *Select > Select Lowest*. All the clones sharing the same reference appear with manipulator handles. You can now enter the PEP editor by clicking its button on the PEP palette:



Only the last clone you selected appears in the PEP editor, but all clones change with each edit.



PEP modeling cloned houses. Note that pushing up points occurs on all the clones, not just the clone in the PEP editor. You must choose Select Lowest before PEP editing clones.

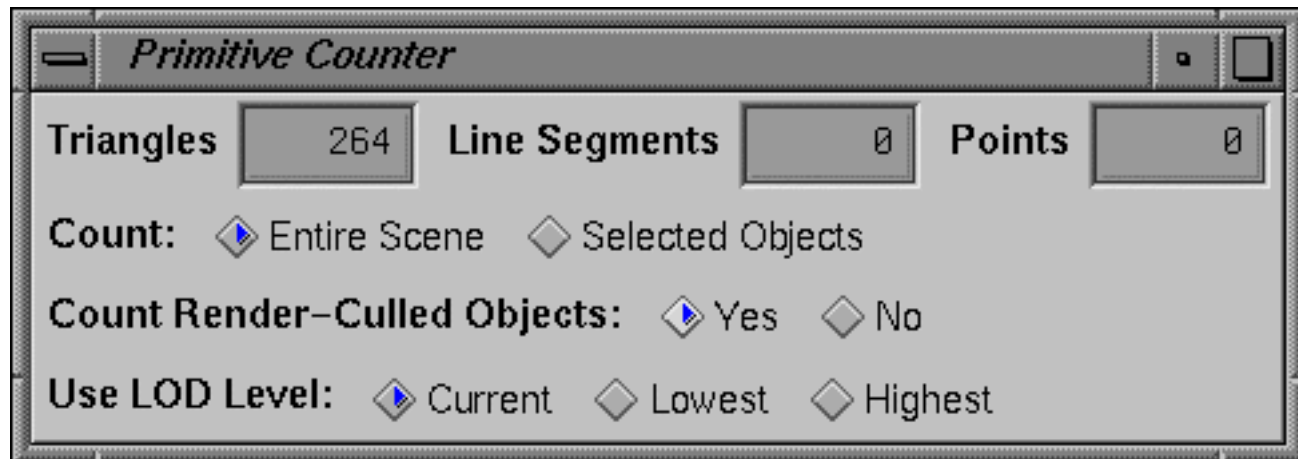
Grouping Clones

You can group a clone to a non-clone object. However, any changes in appearance that you make to a group that includes a clone affects all the clones. If you select a clone not in the group, the appearance changes affect the grouped clone, but not the other objects in its group.

Checking the Polygon Count

The Primitive Counter lets you check the polygon count for individual objects or all the objects in your scene. You can specify which LOD objects to count for best and worst case viewing scenarios.

Find it: *View > Display Polygon Count*



Triangles, Line Segments, and Points: Displays the number of triangles, line segments, and points in the scene or object as specified by *Count*, *Count Render-Culled Objects*, and *Use LOD Level*.

Count: *Entire Scene* specifies that all the objects in the scene be included in the count. *Selected Objects* specifies that only those objects currently selected be included in the count. *Count Render-Culled Objects* and *Use LOD Level* further limit this count.

Count Render-Culled Objects: *Yes* specifies that only those objects currently visible in the scene should be counted. *No* specifies that objects not currently visible also be counted.

Use LOD Level: *Current* specifies that only the LOD objects visible in the scene be counted. *Lowest* specifies that only the lowest LOD objects (visible or not) be counted. *Highest* specifies that only the highest LOD objects be counted.

Tip: To find the worst-case scenario polygon count for your scene, select *Count: Entire Scene*, *Count Render-Culled Objects: No*, *Use LOD Level: Highest*.

Creating Inline Objects

on this page: [creating](#) | [putting an inline back into the scene](#) | [changing the URL](#) | [bounding box fields](#)

Use inline objects to break a large file into smaller pieces that can be downloaded separately. An *inline* object contains the URL for another file, either on the local system or anywhere else on the Web. Some browsers display bounding boxes for inline objects until the complete inline file is loaded. This technique allows users to navigate through a world while they are waiting for all the pieces to load. For complex files, inline objects are an efficient way to manage data for a large number of objects.

Find it: Click the Inline Editor button on the *Action* palette:



Creating an Inline Object

To create an inline object, follow these steps:

1. In the main window, select the object that you want to make into an inline object.
2. In the Inline Editor, click the *Convert Master Selection to Inline File* button.
3. The File Selection dialog box appears. Type the complete path and filename for the new inline file. VRML files usually end in the suffix *.wrl*.

When you click *OK*, the selected object is no longer shown in the [Outline Editor](#) view of the scene file. Instead, an Inline node is added to the file, with a URL that refers to the inline object. When the scene is displayed, there is no discernible difference between an inline object and any other object in the file, unless you enable the *Display Inline as Bounding Box* feature. On some browsers, when a large scene is downloaded, inline objects are initially displayed as bounding boxes, since they are fetched separately from the main file.

4. Check any of the three boxes if desired:

- **Do you want to display a bounding box in place of the inline object?**
- **Do you want to allow editing of the inline?** If this box is checked, you can edit the inline object or any of its children. When editing is enabled, the Outline Editor's view of the file displays the actual contents of the inline file. If this box is not checked, the Outline Editor shows only a *url* field and the bounding box fields of the Inline node, and you can't edit the contents of the Inline object. In either case, when the file is saved, the inline object in the file is replaced by a URL reference to a file containing that object.

- **Do you want the inline's bounding box fields to be updated when the object is saved?** Be sure to uncheck this box if you specify a bounding box explicitly in the file and you do not want these values overridden.

Putting an Inline Object Back into the Scene

To put an inline object back into the scene (so that it's part of the main file again):

1. In the main window, select the inline object.
2. In the Inline Editor, click the *Fold Inline Back into Scene* button.

The Inline node is removed from the file, and the object or group is reinserted directly into the file.

Changing the URL

You can change the URL for the inline object by clicking the *Change URL* button to browse the files on your system. The file suffix for VRML files is *.wrl*.

Bounding Box Fields

If you examine an Inline node using the Outline Editor, you'll see it has two fields relating to the object's bounding box: *bboxCenter* and *bboxSize*. Some browsers use these fields to optimize rendering and downloading. These browsers can use the bounding box fields to determine quickly whether the inline objects are in view. If they aren't, the browser doesn't need to fetch the inline file at all. You can specify your own bounding box values using the Outline Editor, or you can allow Cosmo Worlds to fill in the fields for you.

Jump to:

- [Optimizing the Scene](#)
- [Importing Objects as Inlines](#)

Importing Objects as Inlines

Find it: Choose *File > Import as Inline*

When you import an object as an inline, a reference to the file containing the object is added to your scene file. The data within that file is not copied directly. Using inlines reduces file size, but it also places added responsibility on you for managing all the files you use as inlined references.

Jump to:

- [Importing Objects \(general\)](#)
- [Using Inlined Objects \(optimization\)](#)

keep the original, copy and paste (*Ctrl-c*, *Ctrl-v*) the object and select the copy before opening the Polygon Copier. The Polygon Copier is editing the current selection, not creating a new object based on the original--the two windows are for previewing changes and do not reflect two separate objects in the scene.

The Polygon Copier's windows use the [Examiner Viewer](#) by default (its controls are hidden). *Alt*-drag to reposition your view in a window. The view in each window is ganged, so moving your view in one window affects both windows.

Options

The following options affect your edits in the *left* window:

- **Copy Triangles:** Select triangles to copy by clicking them in the left window. Selected triangles appear red.
- **Point-by-Point Triangles:** Click to define three points of the triangle.
- **New Tri Strip:** Define a new triangle strip. Click points to define the triangle. A red, dotted line shows you the selected area as you define it.
- **New Tri Fan:** Define a new triangle fan. Click a point and drag--the dotted line fans out to include existing points.

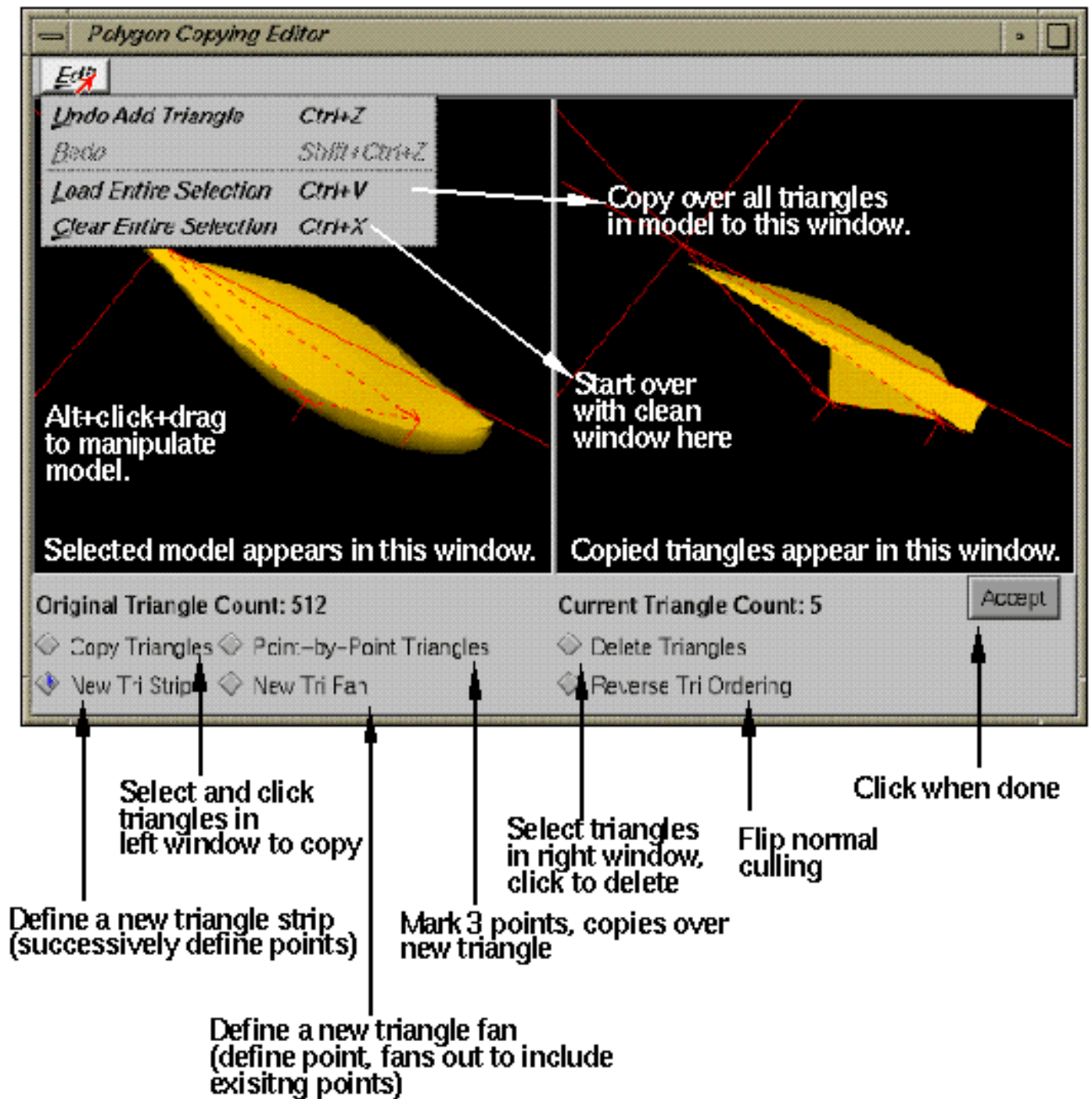
The following options affect your edits in the *right* window:

- **Delete Triangles:** When this option is *on*, clicking triangles in the right window deletes them.
- **Reverse Tri Ordering:** When this option is *on*, clicking triangles in the right window flips their normal culling. This option is useful for fixing normals that appear backward. For example, a negatively scaled object may produce backward normals.

Edit Menu

- **Undo Add Triangle and Redo:** The Polygon Copier has its own undo and redo so you can undo mistakes within the editor window.
- **Load Entire Selection:** Copies over all triangles in original to right window.
- **Clear Entire Selection:** Clear the triangles from the right window.

Jump to: [Optimizing Your Scene](#)



Normal Doctor

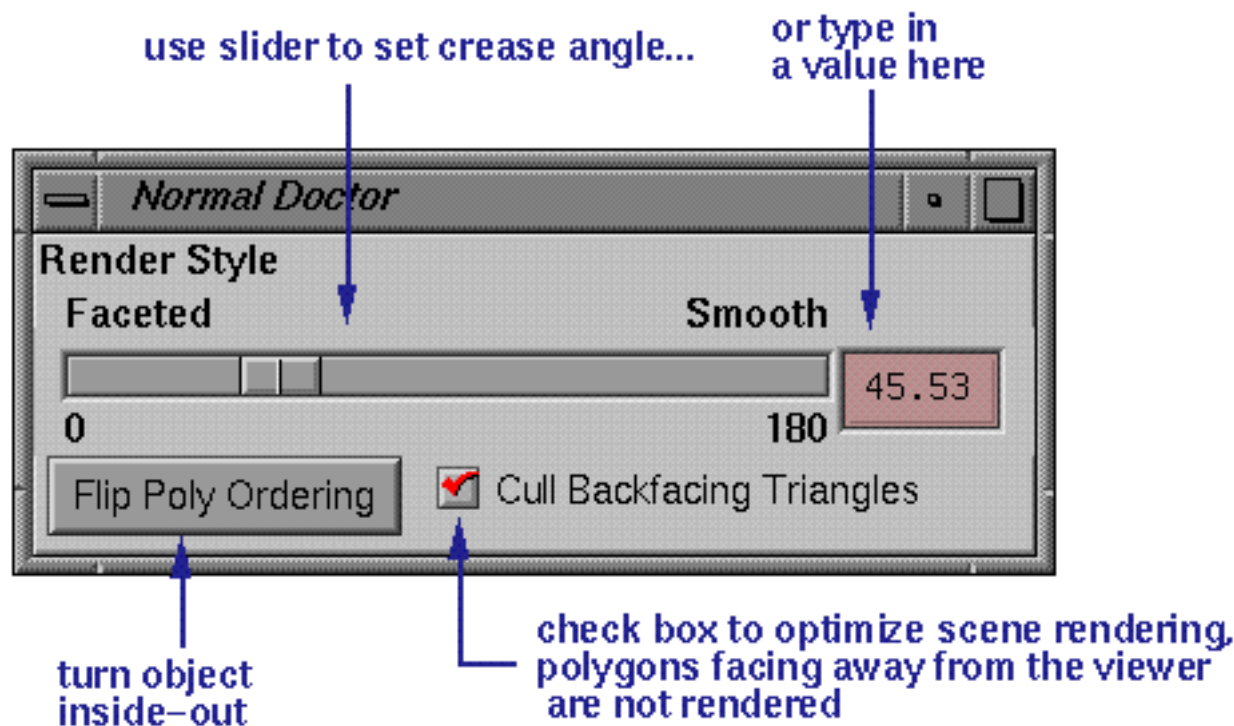


Find it: Click its button on the *Looks* palette:

Use to:

- Change the shading of an object's surface so that it appears more faceted or more smooth. The more smooth an object is, the longer it takes to render in a scene.
- Flip polygon ordering so the object looks "inside-out."
- Cull backface triangles so that polygons facing away from the viewer are not rendered. This can make the object render more quickly in a scene.

How it works:



Notes:

- The values 0-180 refer to the [crease angle](#) in degrees. If the angle between the normals for two adjacent polygons (the dihedral angle) is less than the crease angle, then the polygon will be rendered smoothly. If the dihedral angle is greater than the crease angle, then the polygon will be rendered as faceted.
- If you specify normals with a *Normal* node and *normalPerVertex*, the *creaseAngle* is ignored. So if you select an object which has specified normals, then open the Normal Doctor and set

Render Style, no change will take place.

Jump to:

- [Optimizing Your Scene](#)
- *Related:* [Mirroring and Reordering Polygons](#)

Mirroring and Reordering Polygons

If you have a symmetrical object, it's often easier to construct one half of the object, then flip and paste a copy onto the original half to make the whole shape.

The following example uses a simple shape to illustrate the principles of mirroring. To try it yourself, first create a cone, turn it into a PEP object, then enter the PEP editor. See [PEP Modeling](#) if you don't already know how to do those activities.

1. Once you've got the cone in the PEP editor, select all its polygons. (See [Selecting Polygons](#) for more information.)
2. Copy the polygons by pressing *Ctrl-c* or by choosing *Edit > Copy* from the menu bar.
3. Place the PEP Jack in the scene at the base of the cone. To do this, press the PEP Jack button on the PEP palette:

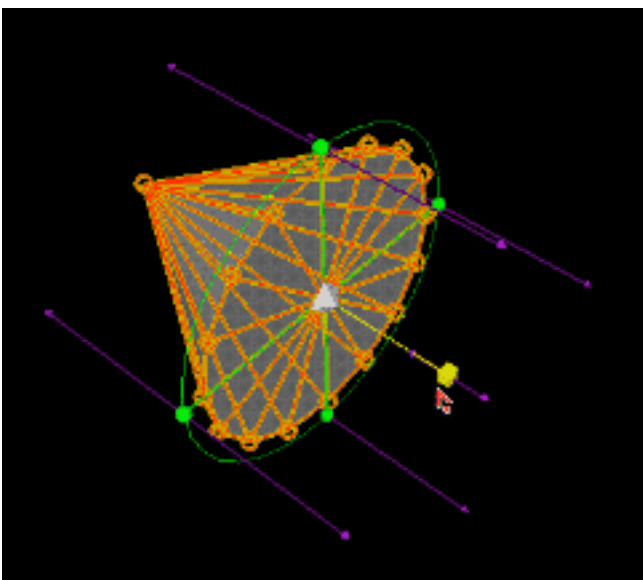


Drag the mouse in the scene. An outline of the PEP Jack follows the cursor. Release the mouse to place the PEP Jack. You can also click and drag the PEP Jack after you've placed it.

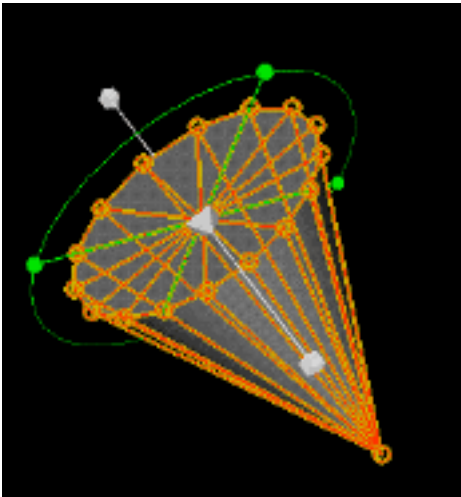
4. *Shift*-drag the PEP Jack's scale handle (the white cube) toward the tip of the cone.

Tip: If you're having a hard time selecting the white cube, try this. Place the mouse cursor over the scale handle. When the cube turns orange, hold down the right mouse button, then press *Shift*. If you accidentally deselect the polygons, press *Ctrl-z* one or more times to undo your mistakes.

The PEP Jack displays purple arrows indicating a constrained scale:



5. Continue to *Shift*-drag past the cone's bottom plane. This inverts the cone, as shown here:



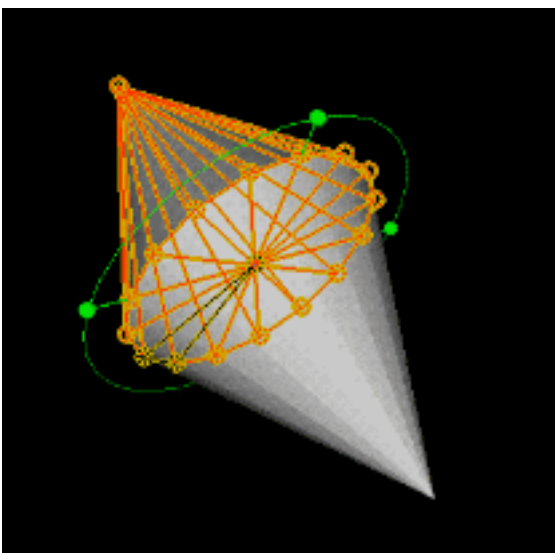
This movement also results in a negative scale--the inverted cone's coordinates become negative and the vertices have a different ordering from the original. This causes a problem which you'll fix in the next step.

6. While the inverted polygons are still selected, press the Reorder Polygons button, found on the PEP palette:

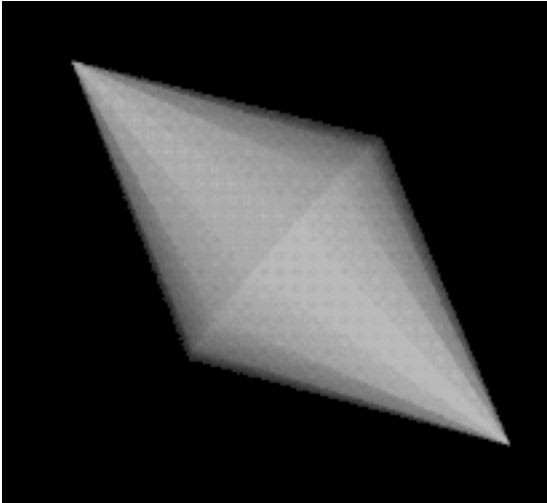


This fixes the negative scale problem described in the previous step. If you don't reorder the polygons after pasting back the original, you'll have two objects with different orderings. The result is that the polygons in one of the cones are culled the wrong way, so you see the inside of the cone.

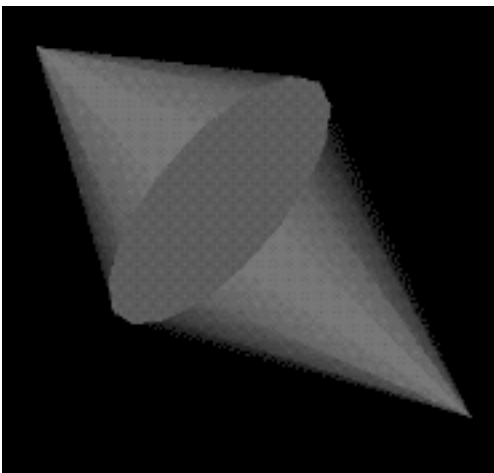
7. Paste the original object in it's original position by pressing *Ctrl-v* or by choosing *Edit > Paste* from the menu bar. The cone looks like this now:



8. The final object is a single PEP object consisting of 2 mirrored sides:



The object would have looked like this if you hadn't reordered the inverted cone's polygons:



Selecting Points, Edges, and Polygons

on this page: [task summary](#) | [how to](#) | [feedback](#) | [master selection](#)

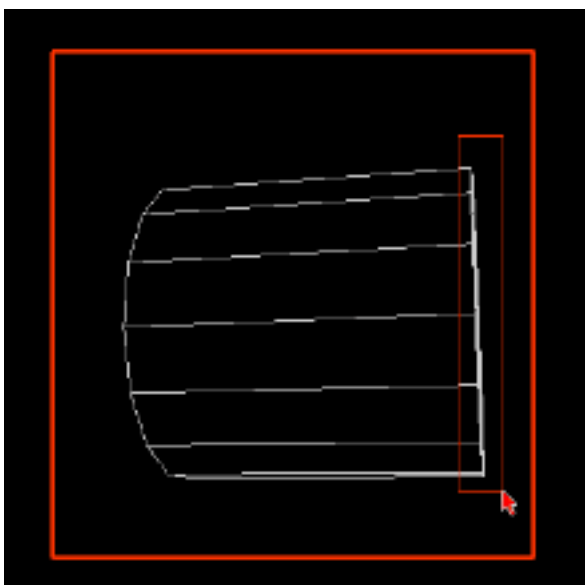
Task Summary

1. Before you can select points, edges, and polygons, make sure you are in [PEP editing](#) mode.
2. Use the most convenient way to select PEPs. See [How to Select PEPs](#) for details.
3. Verify that you have selected the right PEPs. Highlighting feedback can help you pinpoint which PEPs you have selected. You can also use this feedback to pinpoint PEPs you want to select. See [Feedback Guides You](#) for details.

How to Select PEPs

There are a few ways to select points, edges, and polygons:

- Click to select individual PEPs.
- *Shift*-click to select additional PEPs.
- *Shift*-click to deselect individual PEPs.
- **Sweep Selection:** Click outside the object (but still within the PEP editor) and drag to form a red box around multiple PEPs. This is called *sweep selection*. For example, to select all the endpoints of a cylinder, reposition the cylinder as shown and drag the cursor to form a box around the endpoints:



Holding down *Shift* while doing a sweep selection adds to previously selected PEPs.

Make sure you are not moving any previously selected PEPs when you drag. When you select PEPs and then drag on the object, you move the selected PEPs. (See [Translating PEPs](#) for more details.) You should always click on the background within the PEP editor to initiate a sweep selection.

- **Paint to Select PEPs:** Use *Paint to Select PEPs* to select multiple PEPs with a single drag. Press this button:



and drag to select multiple PEPs. Any previously selected PEPs will remain selected. If you decide that you'd like to select additional PEPs after you have finished dragging, press the button again. To deselect multiple PEPs, press the button, then *Shift*-drag. Use *Shift*-click to deselect individual PEPs.

- **Note:** A selection is defined by points, and the selection algorithm connects as many polygons and edges as it can within the area of points you have selected. When you use *Shift* to extend a selection, you can select a point some distance from the initial selection, and any new edges or polygons that are completed by the addition of that point will also become selected. Experiment with selecting PEPs to get the best results. You can use *Shift*-click to deselect unwanted points.

Feedback Guides You

- Selected items in the PEP Editor are always rendered so that you can see selected PEPs when viewing from any angle.
- Points, edges, and polygons appear bright when you pass your cursor over them.
- Selected points and edges appear yellow with a red center (orange).
- The master selection appears yellow with a black center.

Define the Master Selection

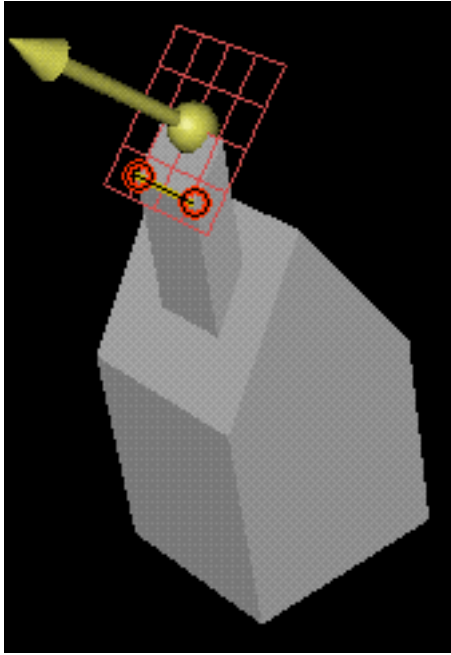
The **master, or primary selection** is the last point, edge, or polygon selected.

Special case: *Sweep selection* defines the master selection as the **point** closest to where you click the cursor when starting the sweep. *Paint selection* defines the master selection as the last **polygon** the cursor passes over.

To redefine the master selection, you may need to deselect a PEP by *Shift*-clicking it and reselect that same PEP (again using *Shift*-click) to make it the most recently selected PEP (and therefore the master selection).

It is important to define which PEPs make up the master selection because the master selection is

used in alignment. In the example below, the chimney is slanted (because its top polygon was extruded and pushed up from the slanted roof). To correctly align the chimney's edges, you need to pick the slanted edge. If the whole polygon or the wrong edge is selected, the edges won't align correctly.



See [Aligning PEPs](#) for more details.

Jump to:

- [PEP Modeling: Editing Points, Edges, and Polygons](#)
- [Tools to Use: Creating and Editing Models](#)

Translating Points, Edges, and Polygons

on this page: [task summary](#) | [position cursor and drag](#) | [example 1](#) | [example 2](#)

Task Summary

1. Make sure you are in the [PEP editor](#).
2. [Select points, edges, or polygons](#) that you want to move.
3. Place the cursor anywhere on the surface of the PEP object and drag. Notice that the selected PEPs move in the direction of your drag, and arrows appear to indicate movement.

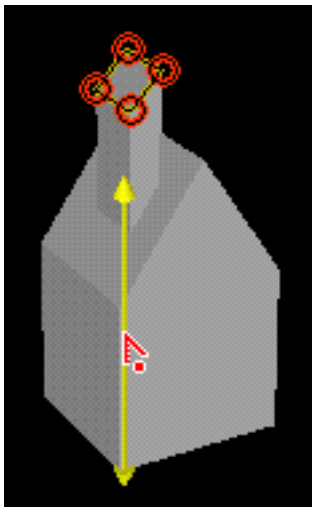
Position the Cursor and Drag

Placing the cursor in the right place before dragging is an important part of moving selected PEPs where you want them to go. Here are some tips:

- **Find an edge or polygon** that exists in the same plane or angle along which you'd like to move the selected PEPs.
- Use [highlighting feedback](#) to find the right plane and angle to drag along. **Pass** your cursor over the surface of an object and notice the highlighting that occurs (**don't click**--if you do, the PEPs you have selected to move will be deselected). You can use this feedback to correctly position the cursor over the desired edge or polygon.
- **Constrain movement:** When selecting an edge or polygon, keep in mind that you can also move perpendicular to that edge or polygon by pressing **Ctrl** while dragging. Pressing **Shift** while dragging constrains movement in a single direction within the plane.
- **Arrow feedback:** When you drag, yellow directional arrows appear to indicate your motion. Dragging produces arrows pointing in four directions. *Shift*-dragging produces a bi-directional arrow, and *Ctrl*-dragging produces an arrow indicating a perpendicular motion (see images below for examples of *Shift*-drag and *Ctrl*-drag arrows).

Example 1

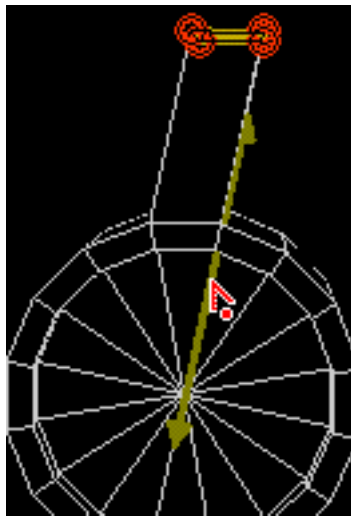
Suppose you want to move the selected polygons to lengthen the chimney on the following house. The chimney's edges should remain parallel with those of the house. Note that neither the plane of the chimney-top nor the perpendicular to that plane points in the direction you want to move. Using highlighting feedback, you would place the cursor over the edge of the house, and *Shift*-drag upward:



Shift-drag up along an edge.

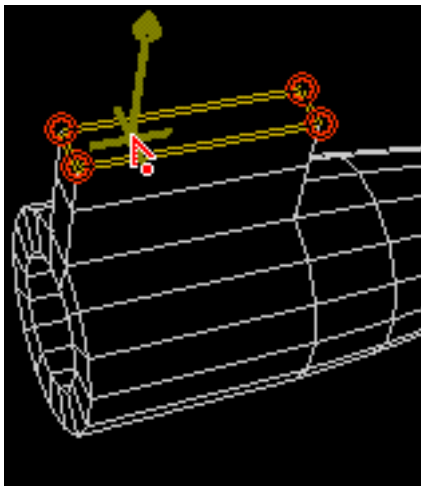
Example 2

Notice the cylinder shape below. (You might recognize it from the Tutorial; see [Shaping the Fin](#) in [Tutorial Part 2: Creating a Rocket](#).) The goal here is to move the selected polygons perpendicular to the cylinder's circle. Suppose you decide to use highlighting feedback to find an edge along which you can guide the movement of the selected polygon. But there is no edge that will give the desired result. If you drag along the highlighted edge, the polygon and its edges appear slanted:



*Choose an edge and **Shift-drag** to constrain the movement in a single direction, but the result is not what you want.*

The solution is to *Ctrl-drag*. Using highlighting feedback, you can ensure that the cursor is correctly positioned over the selected polygon. Then *Ctrl-drag* to constrain the movement perpendicular to the polygon.



*Instead, use **Ctrl-drag** to constrain motion perpendicular to polygon.*

Jump to:

- [PEP Modeling](#)
- [Tools to Use: Creating and Editing Models](#)

Part 2: Creating a Rocket (continued)

Punching in the Back of the Ship

The next task is to make the back end of the ship look indented by extruding, scaling, and pushing points.

1. Open a new file by choosing *File > New*. Then import *shipConical.wrl*, a file provided for this tutorial. Choose *File > Import*. Clear the *Selection* field, then type in this pathname and press *Enter*:

```
/usr/share/Insight/library/SGI_bookshelves/Help/books/  
CosmoWorlds_UG/Models/shipConical.wrl
```

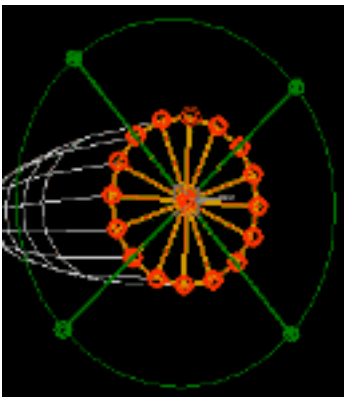
You can save this new file as *rocket.wrl* (the file you've been working on) or give it a new name.

2. Select the shape and enter the PEP modeler by pressing *Ctrl-e*.
3. Reposition your view so you can see the back side of the rocket shape. (*Alt-drag*
4. Activate the PEP Jack by clicking its button on the PEP palette:




Drag it to the center point on the back end of the ship and click to place it there.

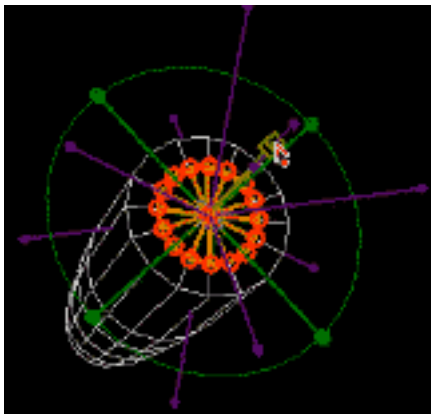
5. Drag a box around the points on the back end of the ship to select them.



PEP Jack placed on center point and endpoints selected.

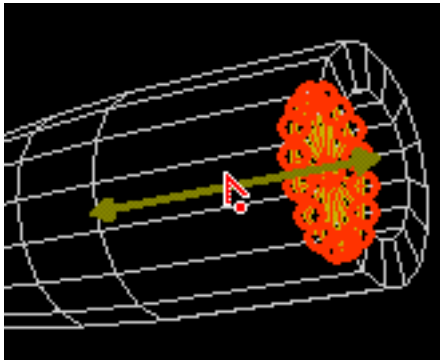


6. Click the Extrude Polygons button , then drag the scaling cube on the PEP Jack to scale in the second ring of points, inside the first.

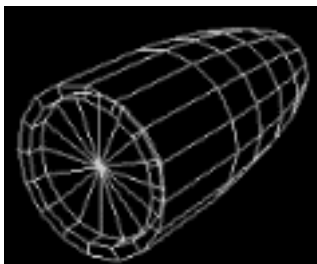


Click Extrude Polygons, then drag PEP Jack handle inward to create a second ring.

7. Put away the PEP Jack by clicking on its button again.
8. Now push the inner circle into the body. Make sure the inner circle of points is still selected. (Use *Ctrl-z* to undo if you've accidentally deselected the points.) Click the Extrude Polygons button. Place the cursor over the side of the cylinder, then *Shift*-drag to the left (along the side of the cylinder) to push the selected points in.



Press Extrude Polygons, then Shift-drag to push the selected points in.



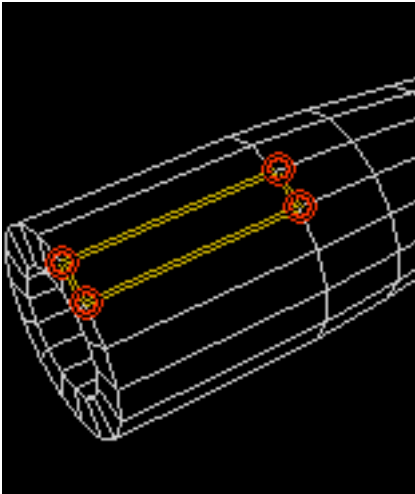
The final product for this section.

9. That's it for the tail end of the rocket! Take a moment to save your file before continuing with *Shaping the Fin*.

Shaping the Fin

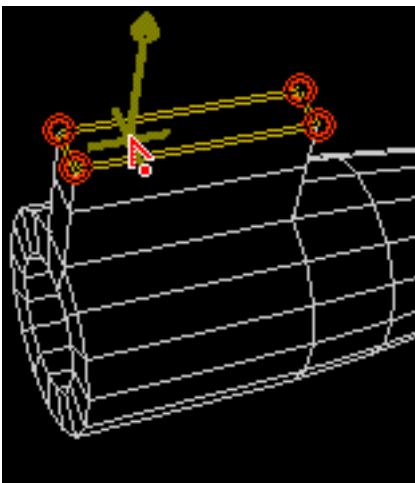
The next task is to extrude and push out segments to create the fin. You've already learned about extruding. This section combines a split function with the extrusion. If you're pressed for time, you can skip this section and learn about coloring the rocket in the next section.

1. Select a long quad on the ship's side, towards the back end.



Click to select a polygon.

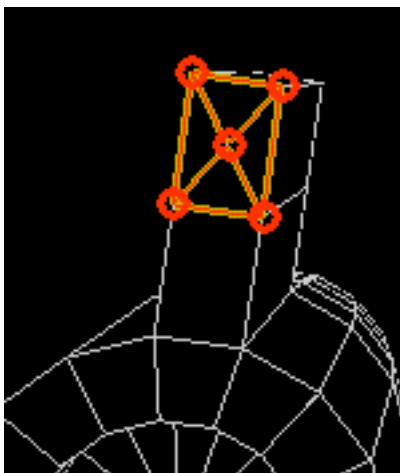
2. Click the Extrude Polygons button, place the cursor over the top of the selected polygon, then *Ctrl*-drag to pull out a new segment. Repeat to add a second section (press Extrude Polygons, *Ctrl*-drag).



Press Extrude Polygons, then Ctrl-drag up to create a new section. Repeat.

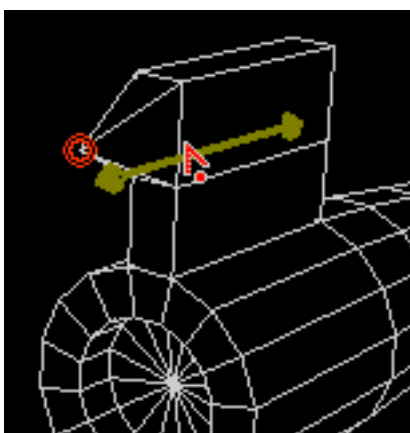
3. Now, select the face of the most recent extrusion that is facing towards the back of the ship. Make a point on the back of the fin by clicking on the *Split Quads in an X Shape* button found on the Split palette:





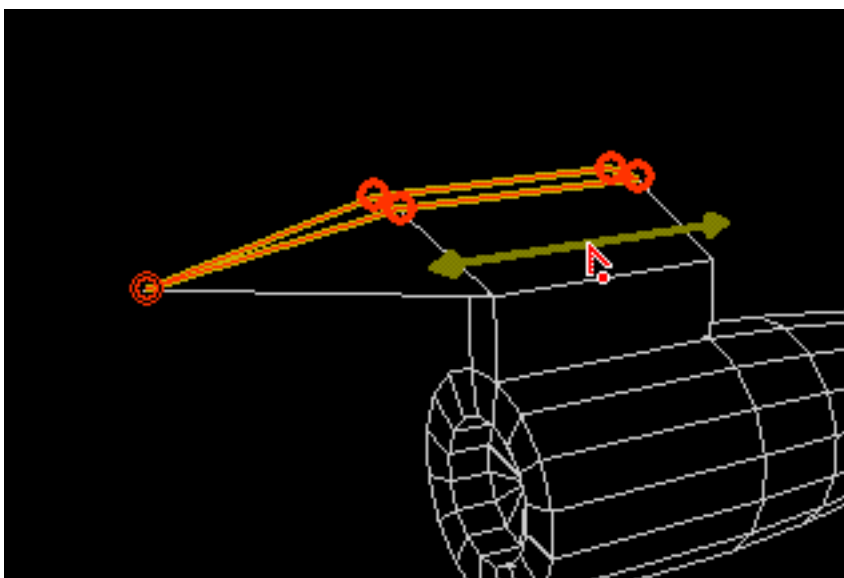
Select polygon and split into X shape.

4. Click on the background to deselect the quads, and click on just the center point to select it. Place the cursor on the side of the quad, and *Shift*-drag to the left to push out the point.



Select middle point of X, then Shift-drag to push point out.

5. Looking at the ship sideways, sweep-select all the top points of the fin to select them. Use *Shift*-sweep to select any missed points. Then *Shift*-drag along a side and move the selected points to the left.

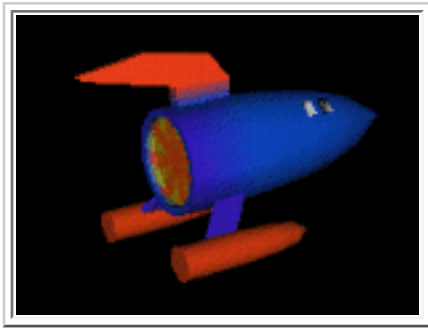


Select polygons making up top of fin. Shift-drag along side to elongate.

6. That's it for the fin! Next, you'll color the rocket. Save your file before continuing.

[Continue](#) 

Part 2: Creating a Rocket



In *Part 2: Creating a Rocket*, you'll learn how to model a rocket by using the PEP modeling tools. PEP refers to the individual points, edges, and polygons of an object. Editing PEPs in the PEP modeler is a powerful feature of Cosmo Worlds. As you'll see, a single shape can be modeled into a more complex creation.

After using the PEP modeler, you'll use layout tools to assemble the finished rocket ship.

Part 2 contains the following sections:

- [Starting Out](#) ↗
- [Adding New Sections to the Ship's Basic Shape](#) ↗
- [Scaling PEPs to Create the Nose of the Ship](#) ↗
- [Punching in the Back of the Ship](#)
- [Shaping the Fin](#)
- [Coloring the Rocket](#)
- [Importing and Placing the Pontoon and Fin](#)
- [Grouping the Rocket](#)
- [Saving and Closing the File](#)

Starting Out

By now, you should have created a file called *planet.wrl* that contains a sphere with a texture applied so that it resembles a planet. If you haven't done this, see [Part 1: Making a Planet](#).

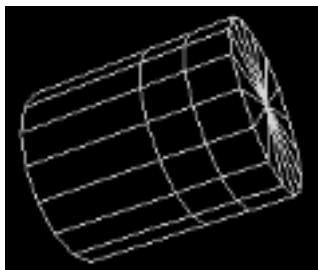
If you have completed Part 1, continue here:

1. If you still have Cosmo Worlds open from *Part 1: Making a Planet*, make sure you have saved *planet.wrl*. If you exited the program after finishing Part 1, then open Cosmo Worlds now.

2. Create a new file by choosing *File > New*, then save as *rocket.wrl*.
3. Import *shipBasic.wrl*, a file provided for this tutorial. Choose *File > Import*. Clear the *Selection* field, then type in this pathname and press *Enter*: **`/usr/share/Insight/library/SGI_bookshelves/Help/books/CosmoWorlds_UG/Models/shipBasic.wrl`**

A cylinder appears.

4. Hold your mouse over the scene and press the right mouse button to see the Viewer menu. Choose *Draw Style > Hidden Line* to see the model's segments:



Note: You can always undo a mistake by pressing *Ctrl-z*.

Adding New Sections to the Ship's Basic Shape

Your first task is to add segments to the ship's basic shape (a segmented cylinder). You'll do this in the PEP modeler, a mode for editing the points, edges, and polygons of a PEP object.

Jump to: [PEP Modeling: Editing Points, Edges, and Polygons](#) for more information on this topic.

1. Make sure your are in pick mode (the arrow is selected on the viewing toolbar).
2. Click the model to select it, then press the *Convert to PEP Object* button on the PEP palette:



Notice at this point that most of the buttons on the PEP palette are grayed out.

Note: If this button is grayed-out, the model is already converted. Go to Step 3.

3. Click the *Select PEPs* button on the PEP palette to enter PEP editing mode:

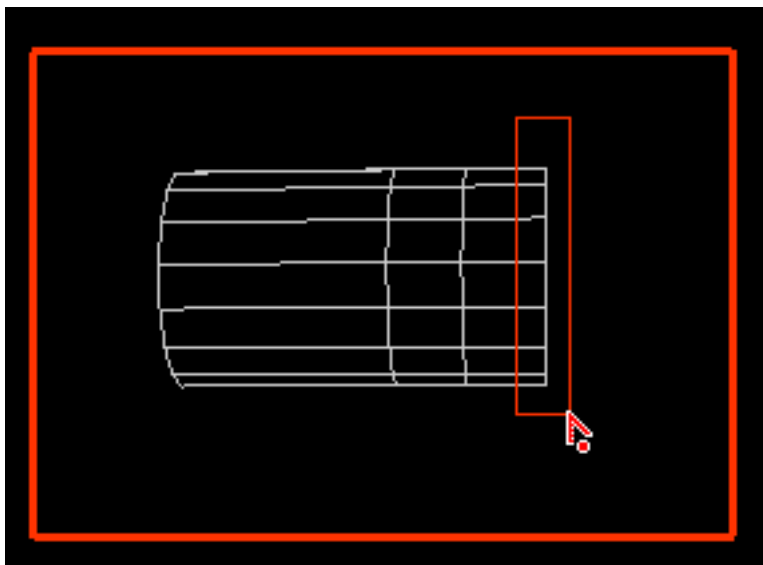


Notice that most of the buttons on the PEP palette (and the Split palette, if it's open) become available to you (they are no longer grayed out). An orange box appears around the cylinder to

indicate PEP editing mode. PEP objects can be edited at any time by selecting them and pressing the Select PEPs button. If you click outside the orange box, you will no longer be in PEP editing mode.

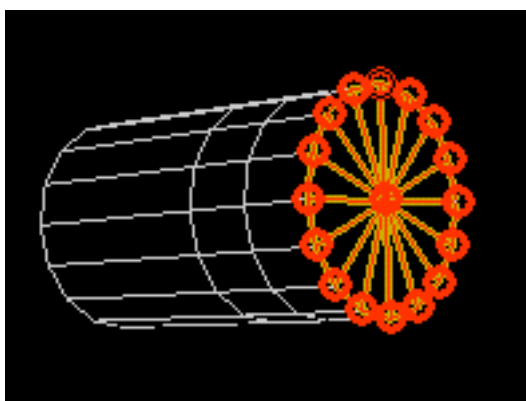
Shortcut: Select the cylinder and press *Ctrl-e* to enter the PEP modeler.

4. Next, select all the polygons on one end of the cylinder. Reposition your view of the cylinder so that you can see its side (*Alt-drag*), then drag to form a box around the end of the cylinder. This is called *sweep-selection*.



Drag the cursor to select endpoints.

5. Reposition your view again so you can verify that all the endpoints are selected, and that you haven't selected other parts of the model:

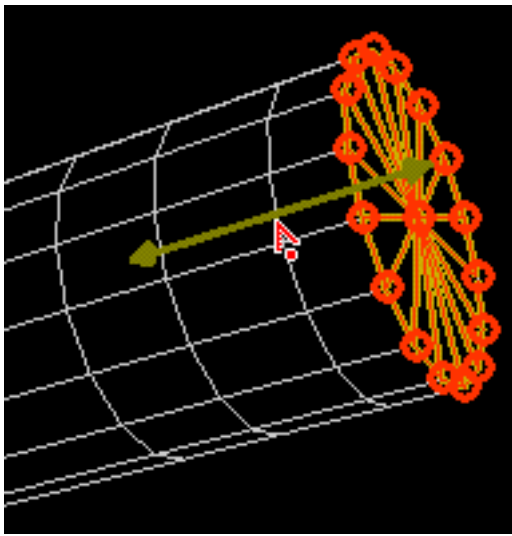


Verify that only the endpoints are selected.

6. With the endpoints still selected, click the *Extrude Polygons* button on the PEP palette:

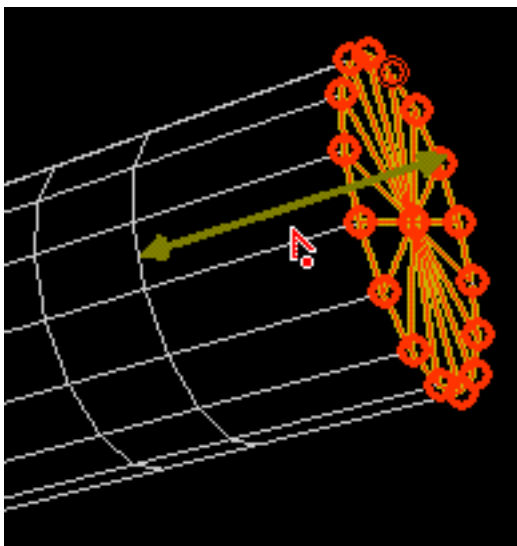


Place the cursor over the side of the cylinder (as shown below), and *Shift-drag* parallel to the side to push up a new segment about the same size as the other short segments. *Shift* constrains the drag in a single direction and is indicated by a bi-directional arrow.



Click the Extrude Polygons button and add a new, short segment.

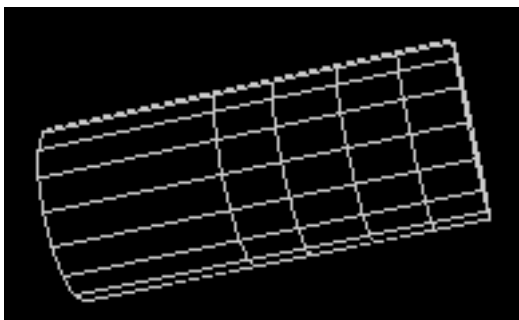
Notice the difference between extruding and moving selected PEPs (as shown above), and just moving PEPs by dragging them and not pressing the Extrude button:



This is what happens if you forget to press Extrude--you enlarge the same segment instead of creating a new one.

7. Add another short segment to the cylinder as described in step 6.

If you've followed the steps correctly, you now have a cylinder with one long segment, and four shorter segments:



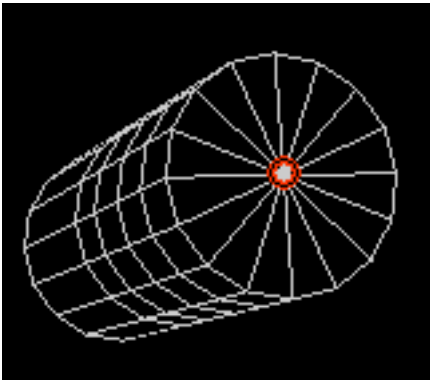
The finished product for this section.

Terrific! You have finished adding segments. Next, you'll scale these segments to form the nose of the rocket. Take a moment to save your file before continuing.

Scaling PEPs to Create the Nose of the Ship

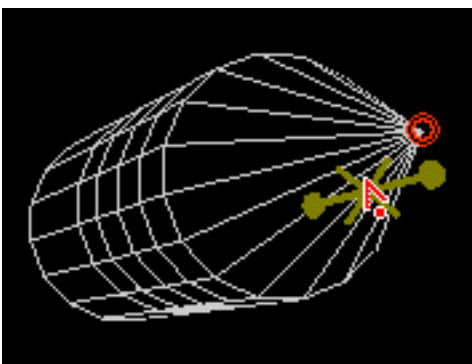
The next task is to make the ship aerodynamic, by scaling down the sections as they get closer to the nose of the ship.

1. Make sure you are still in PEP editing mode. If you are not, select the cylinder and press *Ctrl-e*.
2. Click over empty space to deselect all the endpoints (if they are still selected). Then click on the single point in the middle on the end of the cylinder (the same end at which all the endpoints were previously selected).



Select just the centerpoint.

3. Position the cursor over the top of the cone (the end with the selected point). Use *Ctrl-drag* to push the point out perpendicular to the end. Remember that in the last section, you used *Shift* to constrain movement in a single direction. The *Ctrl* key performs actions opposite to *Shift*.



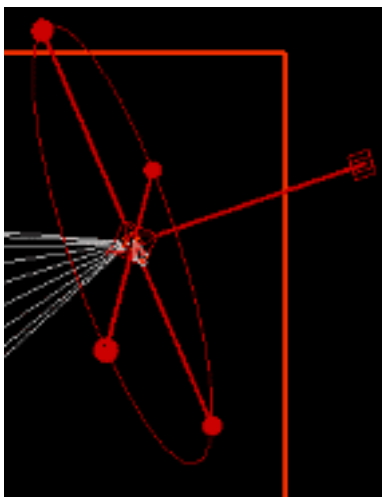
Notice placement of cursor. Ctrl-drag to pull out centerpoint.

This cone is too sharp to make an attractive nose for the ship. In the next step, you'll round out the cone by scaling its points.

4. Activate the PEP Jack, a tool for rotating and scaling points, edges, and polygons, by clicking its button on the PEP palette:

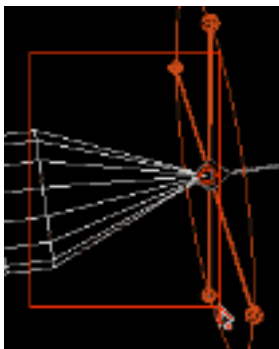


5. The PEP Jack appears "stuck" to your cursor as you pass it over the work area. Drag the PEP Jack and click on the middle point of the cone you just created to place the PEP Jack there. Depending on how close you have dollyed-in, parts of the PEP Jack might appear outside the PEP editing area (outside the orange box). Dolly-in closer to make all the parts of the PEP Jack appear inside the editing area (use the Dolly thumbwheel in the main window or use *Ctrl-Alt-drag* middle).



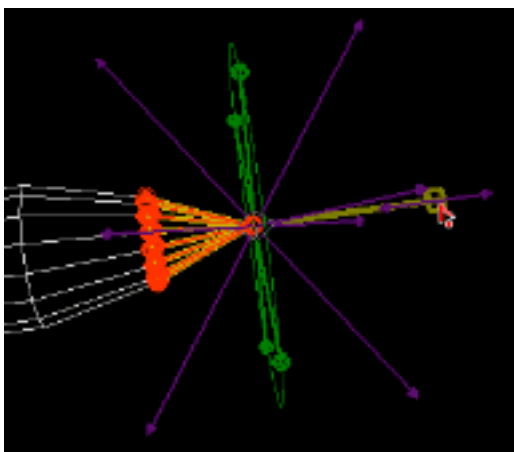
Place the PEP Jack on the selected endpoint.

6. Select all the cone's points by sweep-selecting (dragging a box around them).



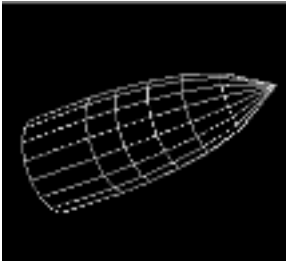
Drag around the cone to select its points.

7. Click and drag on the small cube of the PEP Jack to scale the points into a more round shape:



Drag end of PEP Jack to scale selected points.

8. Repeat selecting and scaling all circles of points except the two nearest the back of the ship. If you'd like to save time, you can skip this last step. In the next section, you'll start out with *shipConical.wrl*, a file with the completed cone shape.



The finished product for this section.

[Continue](#) ➡

[Part 2: Creating a Rocket](#) (continued)

Coloring the Rocket

Now it's time for a little color. You'll use the Color Per Vertex Editor to color the individual polygons of the shape.

1. You should have Cosmo Worlds open with the model of your rocket with its newly created fin. If you don't have your model available, open a new file (*File > New*) and import *shipAndFin.wrl*, a file provided for this tutorial. Choose *File > Import*. Clear the *Selection* field, then type in this pathname and press *Enter*: **/usr/share/Insight/library/SGI_bookshelves/Help/books/CosmoWorlds_UG/Models/shipAndFin.wrl**
2. Change the draw style to *As Is*. In the [Viewer Menu](#), choose *Draw Style > As Is*.
3. Select the ship by clicking on it.
4. Open the Color Per Vertex Editor by clicking on its button in the *Looks* palette:




The Editor window pops up.

5. Notice that a copy of the selected object appears in the editor window. This is to preview the color before pressing *Apply*. Once you press *Apply*, the coloring is applied to the object in the scene. You can undo mistakes in the Color Per Vertex Editor by using *Ctrl-z*.


To reposition your view of the rocket in the editor's preview window, *Alt-drag*.

Also notice the buttons along the bottom of the window. Pass your cursor over the buttons to see a description of what each button does.

For an overview of the icons and buttons, look at a diagram of the [Color Per Vertex Editor](#). 

6. Make the ship blue.



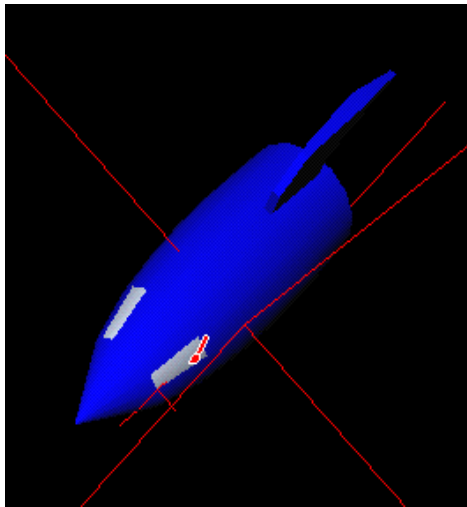
- Press the pick button  and click the ship in the preview window to make the color editor appear.
- Choose a color. Jump to [Color Editor](#) to learn more about using this editor.
- To color all the polygons at the same time, click the button "Edit Color of Every Color in Shape" by choosing the last icon along the bottom of the editor.
- Press *Ctrl* to temporarily switch to the paintbrush, and click on the rocket in the preview window. The rocket turns blue.

7. Make the windows gray.

- Use the paintbrush to select one of the polygons at the nose of the rocket to be a

window.

- Choose the Edit Polygon Face button (the third button from the left).
- Switch back to the pick icon, and change the color in the color editor to gray. The polygon you selected turns from blue to gray.
- Choose a polygon on the opposite side of the nose and color it gray to make a second window.

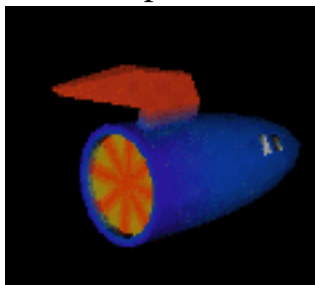


8. Color the fin red.

- Use the paintbrush to select a polygon on the fin.
- Use the pick button to change the color to red in the color editor.
- Use the paintbrush to paint all the polygons making up the fin.

9. Color the rocket's end to a fiery glow.

- Use the paintbrush to select a polygon at the tail of the rocket.
- Use the pick button to change the color to orange in the color editor.
- Use the paintbrush to paint all the polygons orange.
- Now choose the Edit Single Vertex button.
- Use the pick button to change the color to yellow in the color editor.
- With the paintbrush, paint around alternate polygons for a starburst effect.



The finished product.

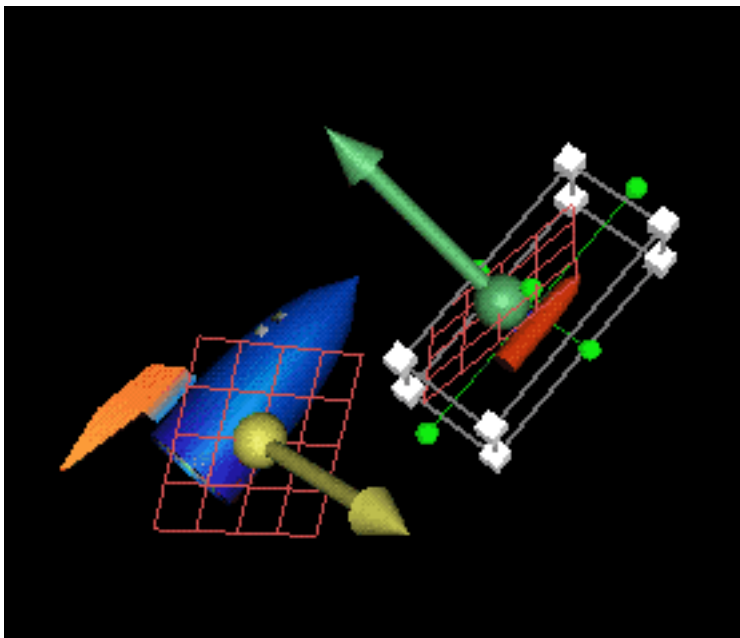
10. Take a moment to save your file before continuing.

Importing and Placing the Pontoons and Fin

The rocket is almost done! Now you'll attach the two "pontoons" to the underside of the ship using

the Snap Target and the Snap Source arrows. You'll use a sample file for this part of the tutorial; it contains a ship with special markings for placing the pontoons. A file with a pre-made pontoon is also included for this tutorial.

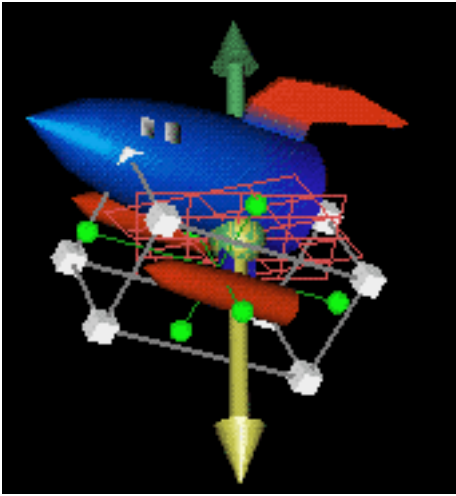
1. If you don't have Cosmo Worlds open with the file you've been working on, then create a new file and import the file *shipAndFinWithColors.wrl*:
`/usr/share/Insight/library/SGI_bookshelves/Help/books/CosmoWorlds_UG/Models/shipBasic.wrl`
- You can save this file as *rocket.wrl* or give it another name.
2. Reposition your view of the ship so you can see its underside. Notice that there are two quads that are darker blue. These are the spots where you will attach the pontoons.
3. In the File menu, turn off *Interactive Import Placement*. Then import the file *shipPontoon.wrl*.
4. De-select the pontoon by clicking on the background.
5. Click-middle over one of the darker blue quads on the ship. The yellow Snap Target appears. You can drag the Snap Target if it isn't precisely where you want it.
6. Select the pontoon and reposition your view so you can see the blue top-end of the pontoon that will be attached to the ship. You may need to hide the ship so that you can see pontoon. Choose *View > Hide Unselected Objects* to hide the rocket (later, choose *View > Show All Objects* to bring the rocket back into the view). *Ctrl-click* middle over the pontoon to place the green arrow for the Snap Source.



The Snap Target and Snap Source arrows placed. Remember to select the pontoon before moving to Snap Target.

7. Re-select the pontoon by clicking on it.
8. Press *Ctrl-m*, or choose *Layout > Move Selection to Snap Target*. The pontoon will snap to the bottom of the ship.

9. Make the second pontoon by cloning the original. Select the pontoon and choose *Edit > Clone*.
10. Place the Snap Target on the other dark blue quad on the other side of the ship. (Click-middle on the quad.)
11. When you paste the clone, it will appear on the Snap Target. Choose *Edit > Paste Clone*.



Jump to: [Layout Tools](#) to learn more about the [snapping arrows](#) and other alignment tools.

12. You're almost done. Now, to group the pontoons to the rocket, then save and close the file.

Grouping the Rocket

1. *Shift*-click to select each part of the rocket.
2. Choose *Edit > Group* to combine the parts into a single object.

Jump to: [Grouping Objects](#) to learn more about this task.

Saving and Closing the File

Yay! You've learned how to select and edit PEPs and how to use the snapping arrows. Your rocket is complete. Choose *File > Save* from the main menu to save your file. If you are continuing to Part 3 now, keep Cosmo Worlds open.

Next, you will create a scene composed of the planet and the rocket. You'll learn how to animate the rocket to land on the planet.

[Part 3: Animating the Rocket](#) ➡

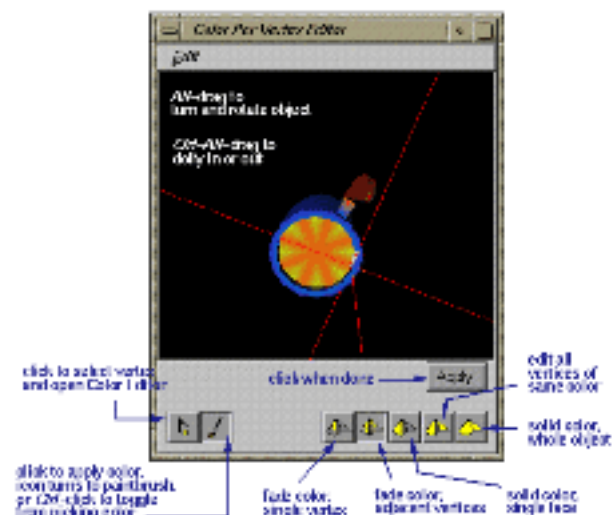
Color Per Vertex Editor

Use the Color Per Vertex Editor to apply color to your models.





Find it: Click from the *Editors* palette.

How it works:



[Click image for full view.](#)

- Drag the cursor over the buttons in the editor for **quick help** (a message appears at the bottom of the editor's window describing what the button does).
- **The Color Per Vertex Editor works with the [Color Editor](#).** The Color Editor appears when you click the arrow icon (Pick mode) and click the object in the Color Per Vertex's preview window.
- Use Pick mode  to select and **load a color** to the color editor. Switch to Paint mode  to **add color** from the color editor. Or use *Ctrl*-click-drag to toggle from Pick mode to paint.
 - Paint works continuously during a drag.
 - Cursor changes to paintbrush in Paint mode.
- If you change color application modes, then you don't need to pick again to start editing colors. You can go straight to the color picker and it affects the object according to the newly selected mode.

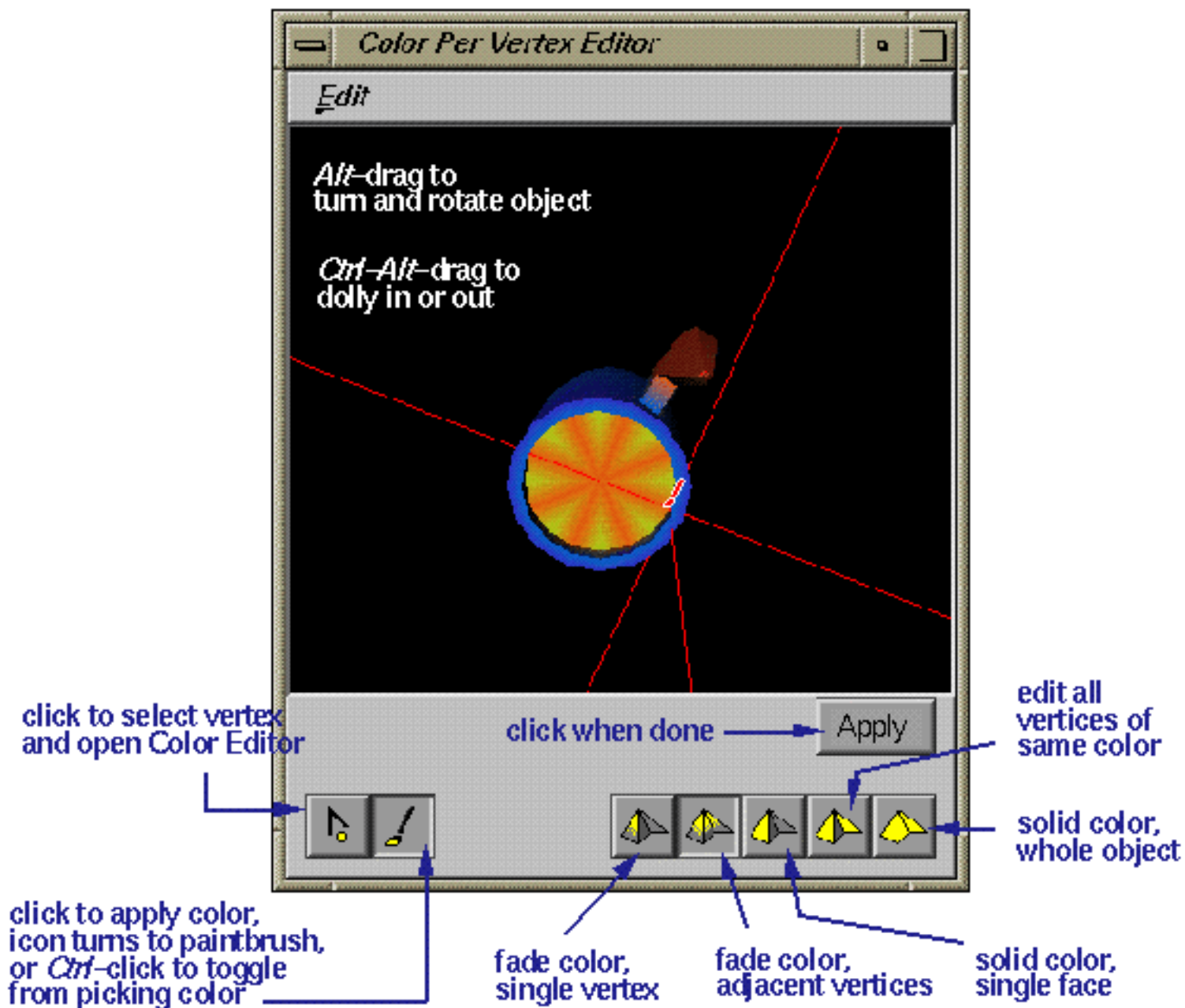
- Use the *Edit* menu to **undo** and redo color changes (or use *Ctrl-z* and *Shift-Ctrl-z*).

Notes:

- The Color Per Vertex Editor converts your models to [PEP objects](#).
- When you click *Apply*, all unused colors and *colorIndex* values are removed to optimize the scene graph.
- If your PEP object has so many polygons that applying color to multiple vertices becomes a problem, consider editing the PEP object so that it has fewer polygons. Use the [Polygon Reducer](#) or the [Polygon Copier](#) for this task.
- See [Coloring the Rocket](#) in the tutorial for an example on using the Color Per Vertex Editor.

Jump to:

- [Applying Color to an Object](#)
- [Changing the Material on an Object](#)
- [Applying a Texture](#)
- [PEP Modeling: Editing Points, Edges, and Polygons](#)



Color Editor

Find it:

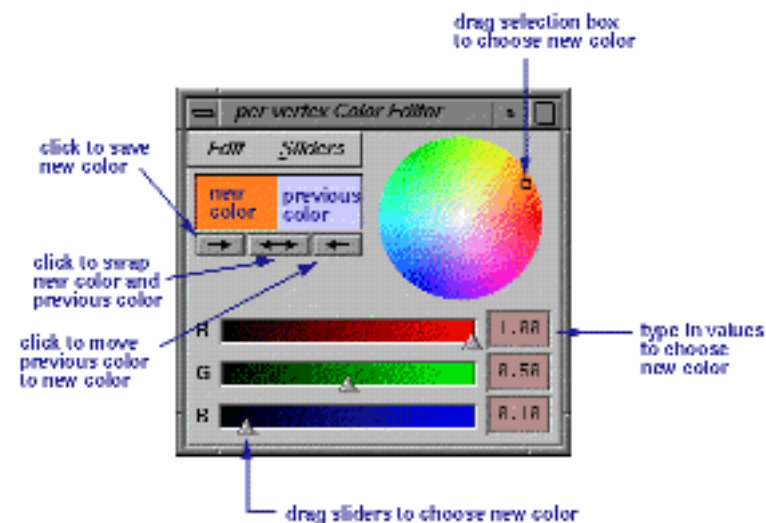
- In the [Material Editor](#), click the boxes next to *Diffuse*, *Specular*, or *Emissive* to bring up the Color Editor.
- In the [Color Per Vertex Editor](#), click the "pick" arrow to bring up the Color Editor:



Use to:

- Edit the Ambient, Diffuse, Specular, and Emissive color in the Material Editor.
- Choose and edit a color in the Color Per Vertex Editor.

How it works:



[Click to see full-size.](#)

- To select a new color, drag the selection box in the color wheel, drag the sliders on the color bars, or type new values in the boxes. The color wheel, sliders, and value boxes are ganged; changing one updates the other two.
- The new color appears in the box to the left of the colorwheel. The previous color is preserved until you save the new color ([see diagram](#)). The saved color then replaces the previous color.
- **Edit Menu**
 - *Continuous*--as you move the slider at the bottom of the window, the target color is

continuously updated.

- *Manual*--lets you experiment with colors. When you're ready to change the target color, click Accept. This setting is useful if redrawing the scene is slow.
- *WYSIWYG*--changes sliders from additive colors displayed to fixed colors. If you are not in WYSIWYG, you create a color by adding or subtracting color from lighter to darker, and mixing the combination of slider values. If you are in WYSIWYG, you pick a color on the bar--there's no adding or subtracting values.
- *Copy*--copies the current color. The color boxes show the current color (left) and stored color (right).
- *Paste*--paste a color that you copied.

- **Sliders Menu**

- *None*--hide all sliders
- *Value*--default, shows value only
- *RGB*--Red, Green, Blue
- *HSV*--Hue, Saturation, Value
- *RGBV*--Red, Green, Blue, Value
- *RGB HSV*--Red, Green, Blue, Hue, Saturation, Value

Jump to:

- [Material Editor](#)
- [Material Palette](#)

Material Editor

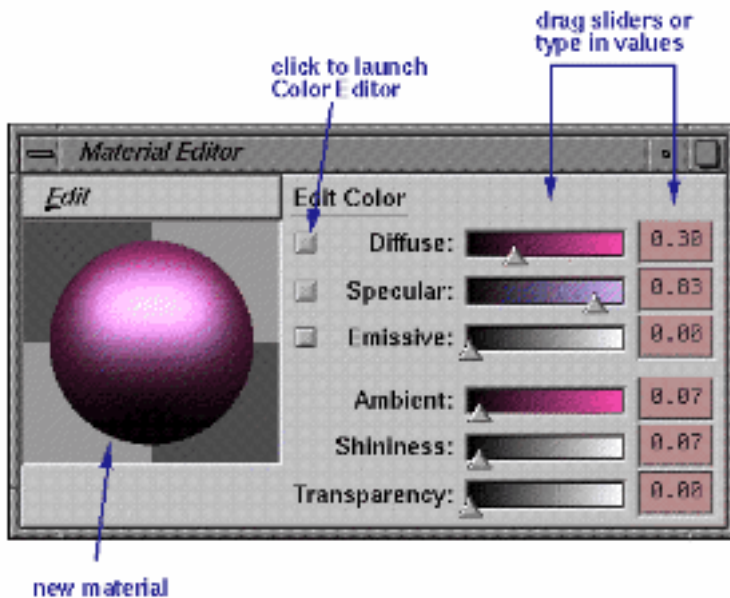
Find it: Click its button on the *Looks* palette:



Use to:

- [Edit an existing material in the Material Palette.](#)
- [Modify a material that's part of an existing palette.](#)
- [Create a new material.](#)

How it works:



[Click image for full view.](#)

- **Edit:**
 - **Continuous** changes the selected model as you edit the material. This is the default setting.
 - **Manual** lets you edit a material without automatically updating the selected model and material palette. Click *Accept* when you're ready to apply the change.
 - **Copy** lets you copy a material in the palette.
 - **Paste** lets you paste a material into the palette.
- **Edit Color:** click the boxes next to *Diffuse*, *Specular*, or *Emissive* to bring up the [Color Editor](#).

- **Sliders:**

- **Diffuse** determines the main color of the material. It also determines whether the material looks rough, or smooth.
- **Specular** controls the amount and color of the specular light. Specular light behaves like a laser beam--the light hits the object and bounces back sharply. As you drag the slider to the right, the material begins to look shiny. As you drag the slider to the left, the material looks flat. Click the box to the left to edit the Specular color.
- **Emissive** determines how much light emanates from the object. As you drag the slider to the right, the material looks as if it's glowing in the dark. Click the box to the left to edit the Emissive color.
- **Ambient** controls how dark or light a material is. Imagine you're in a room with light emanating from all areas of the room. Click the box to the left to edit the Ambient color.
- **Shininess** works with the Specular slider. Together, they determine how shiny or smooth a material appears. The higher the value, the smoother the material appears.
- **Transparency** controls how much light passes through the material. As you drag the slider to the right, the material becomes increasingly transparent.

Creating & Editing Materials

on this page: [edit an existing model](#) | [base a new material on an existing material](#) | [create a new material from scratch](#)

The [Material Palette](#) can use a variety of palettes installed on your system with Showcase. You can create new materials using the [Material Editor](#) and add them to the Material Palette.

Find it: *Looks palette > Material Palette*  or *Material Editor* 

To edit an existing material:

1. Open the Material Palette and click on the material you want to edit.
2. Choose *Edit > Edit Material*. The Material Editor pops up.
3. Use the [Material Editor](#) to edit the material.
4. Save your changes in the Material Palette. (*File > Save* or *Save Palette As*)

To base a new material on an existing material:

1. Open the Material Palette and click on the material you want to copy and edit.
2. Choose *Edit > Copy*, or press *Ctrl+C*.
3. Click on an empty material, and choose *Edit > Paste*, or press *Ctrl+V*.
4. Choose *Edit > Edit Material*. The Material Editor pops up.
5. Use the [Material Editor](#) to edit the material.
6. Save your changes in the Material Palette. (*File > Save* or *Save Palette As*)

To create a new material from scratch:

1. Open the Material Palette and click on an empty material.
2. Choose *Edit > Edit Material*. The Material Editor pops up.
3. Use the [Material Editor](#) to choose a color and set appearances for the new material.
4. Save your changes in the Material Palette. (*File > Save* or *Save Palette As*)

Jump to:

- [Tools to Use: Creating & Editing Models](#)
- [Applying Color to Models](#)

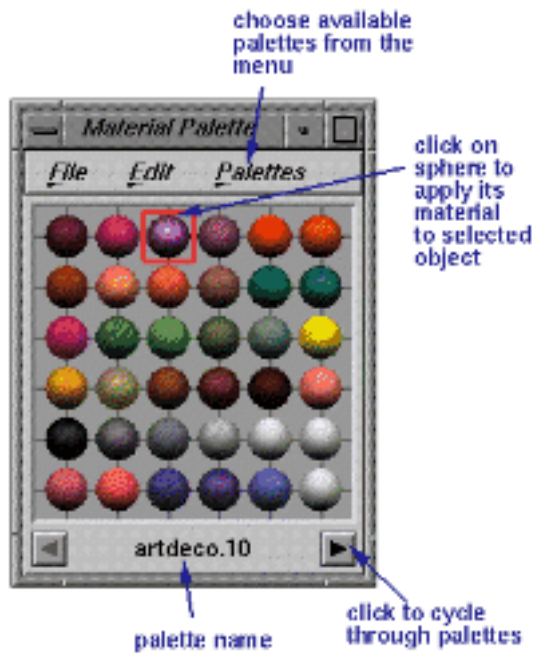
Material Palette

Find it: Click its button on the *Looks* palette:



Use to: Apply a uniform color and appearance to an object.

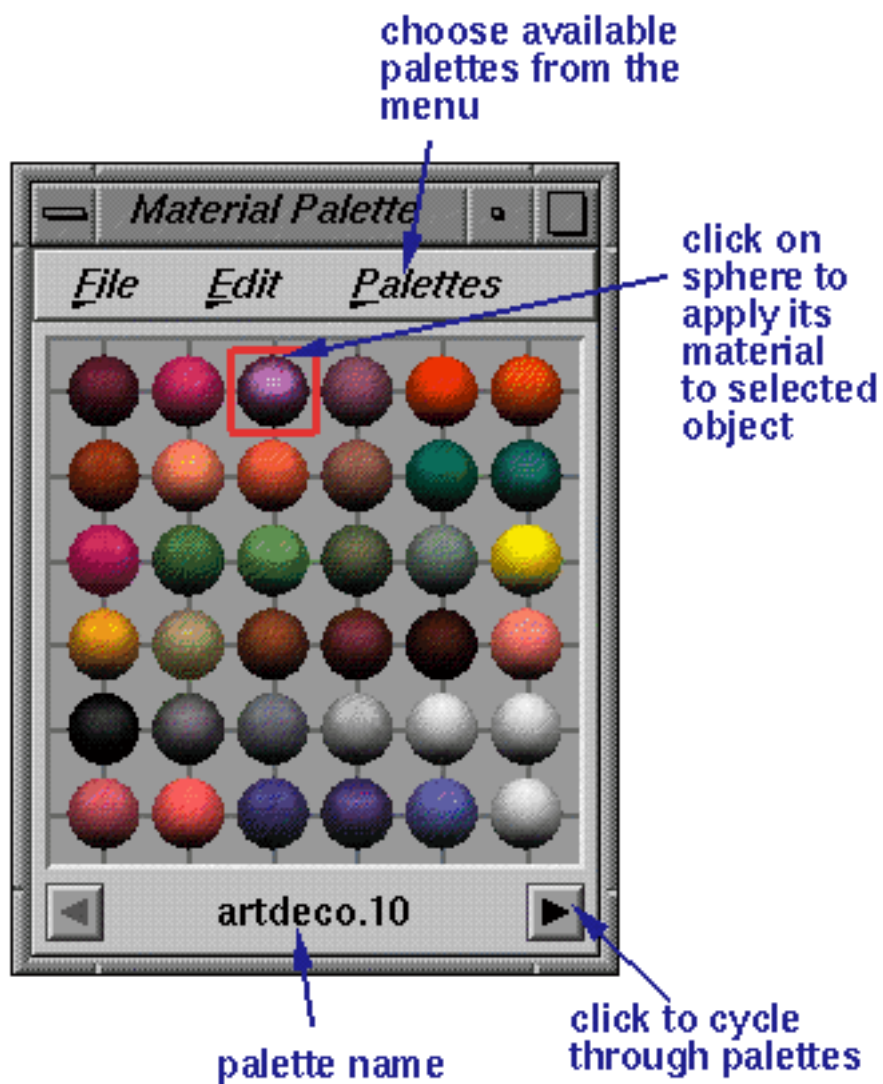
How it works:

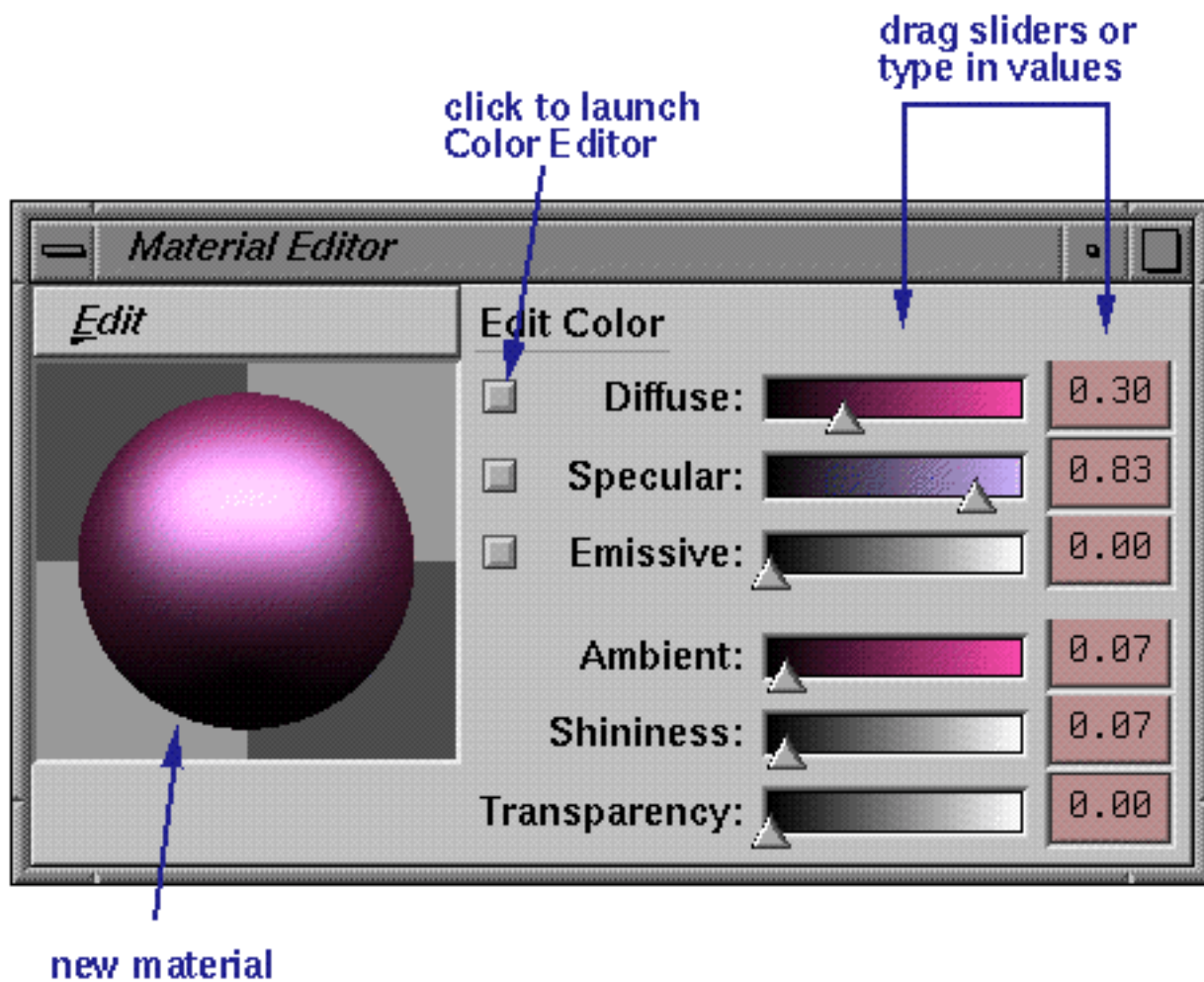


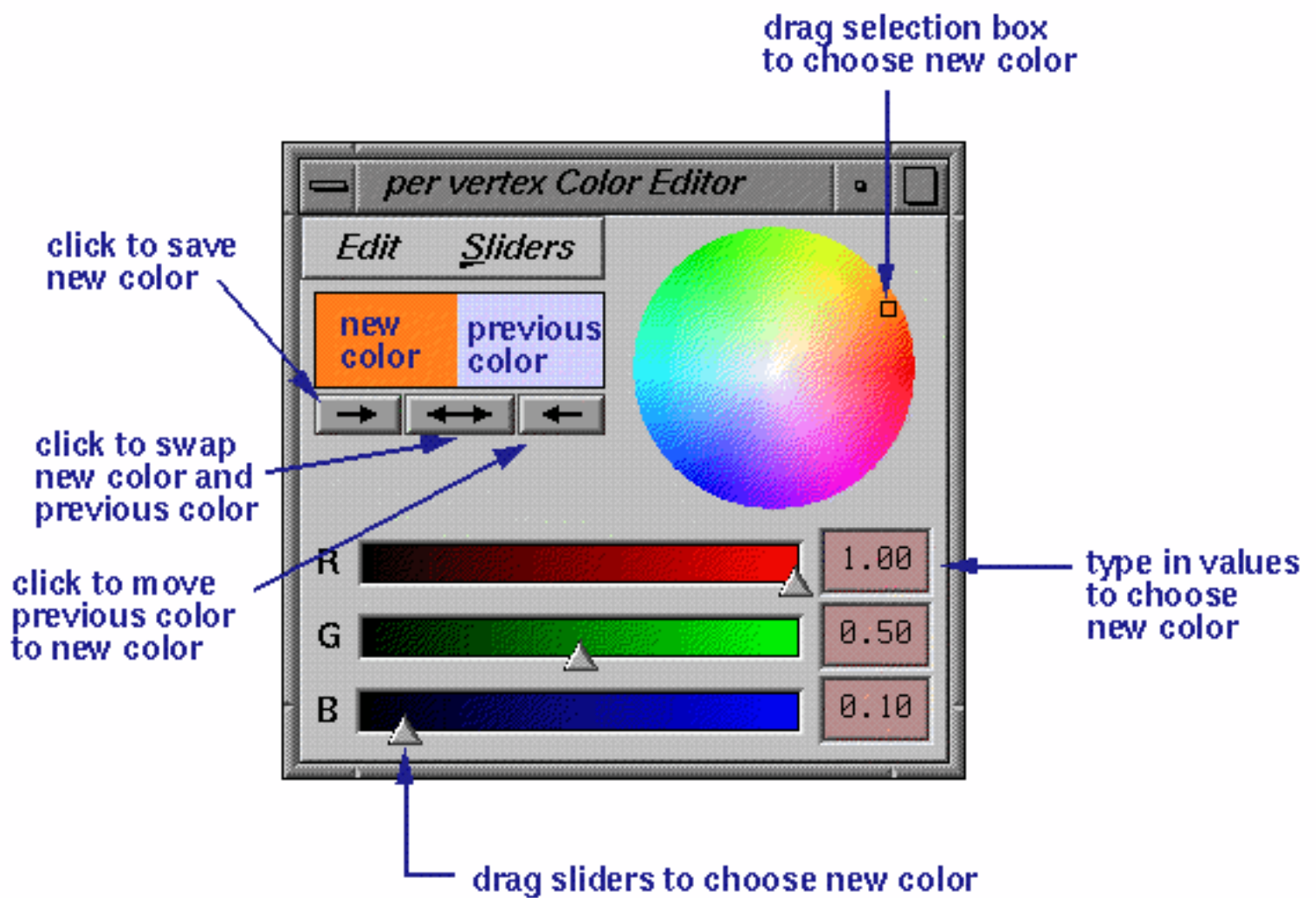
[Click image for full view.](#)

Notes:

- Don't confuse the Material Palette with the [Color Per Vertex Editor](#), which is used to color polygons on a PEP object.
- Use the [Material Editor](#) to create or modify materials and add them to the Material Palette.
- Use *File* to [create and save new palettes](#).
- Use *Edit* to [create or edit materials](#).










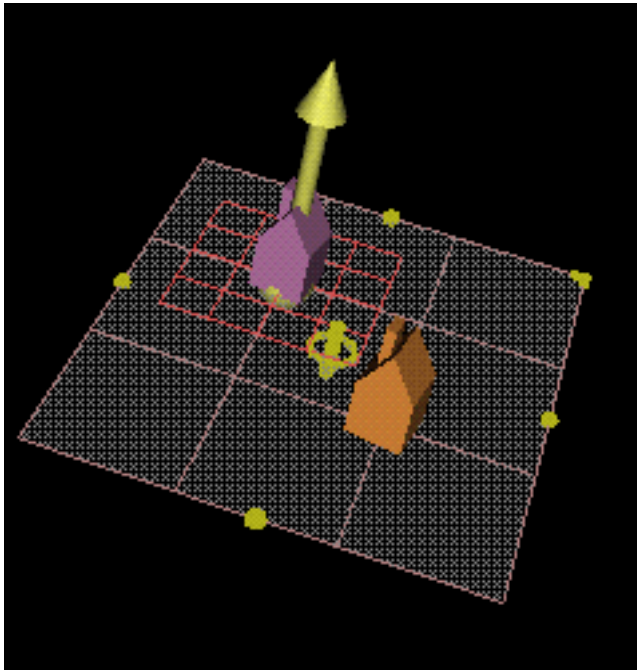


Layout Tools

Find it: press the tool's button on the Layout palette.

- Grid (rectangular grid) 
- Angle (concentric grid) 
- Snap Target 

The [Grid](#)  and [Angle](#)  layout tools work essentially the same way; they provide regularly-spaced reference points for the Snap Target. When you place the target (or source) marker on a construction tool, it snaps to the reference points of the tool. For the Grid, these points are the intersection of the two perpendicular sets of lines. For the Angle tool, these points are the intersections of the rings with the spokes.



Snapping house models to grid.

To lay out some objects along fixed grid points or along a circle, put the target in the correct spot on the appropriate layout tool and move each object to it.

Each tool has interactive handles for rotating, translating, and changing the layout (click the links above to see an explanatory diagram):

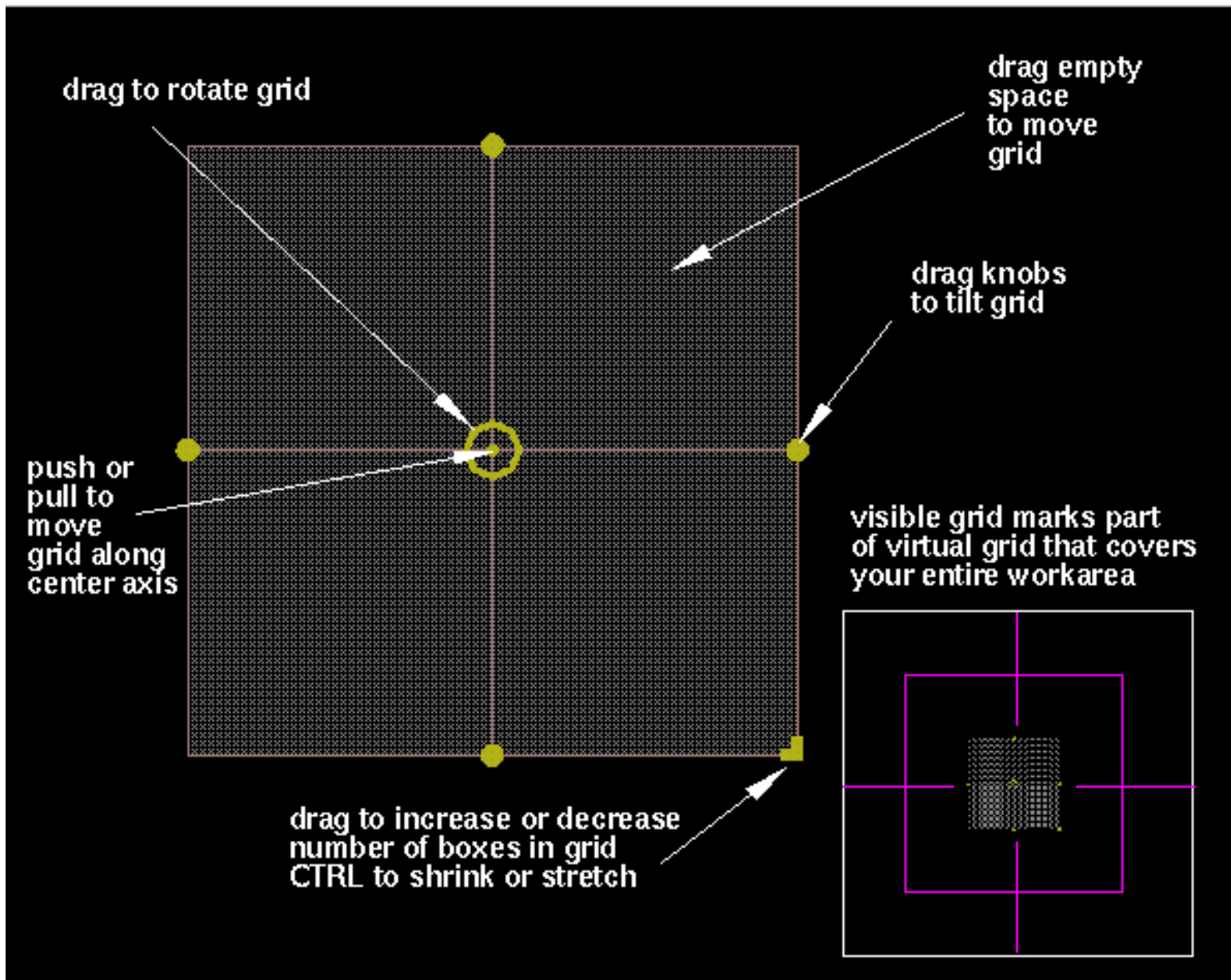
- Drag the grid surface to translate it in the current plane. The Grid tool constrains this to translate in increments of the division size. Press *Ctrl* to unconstrain.

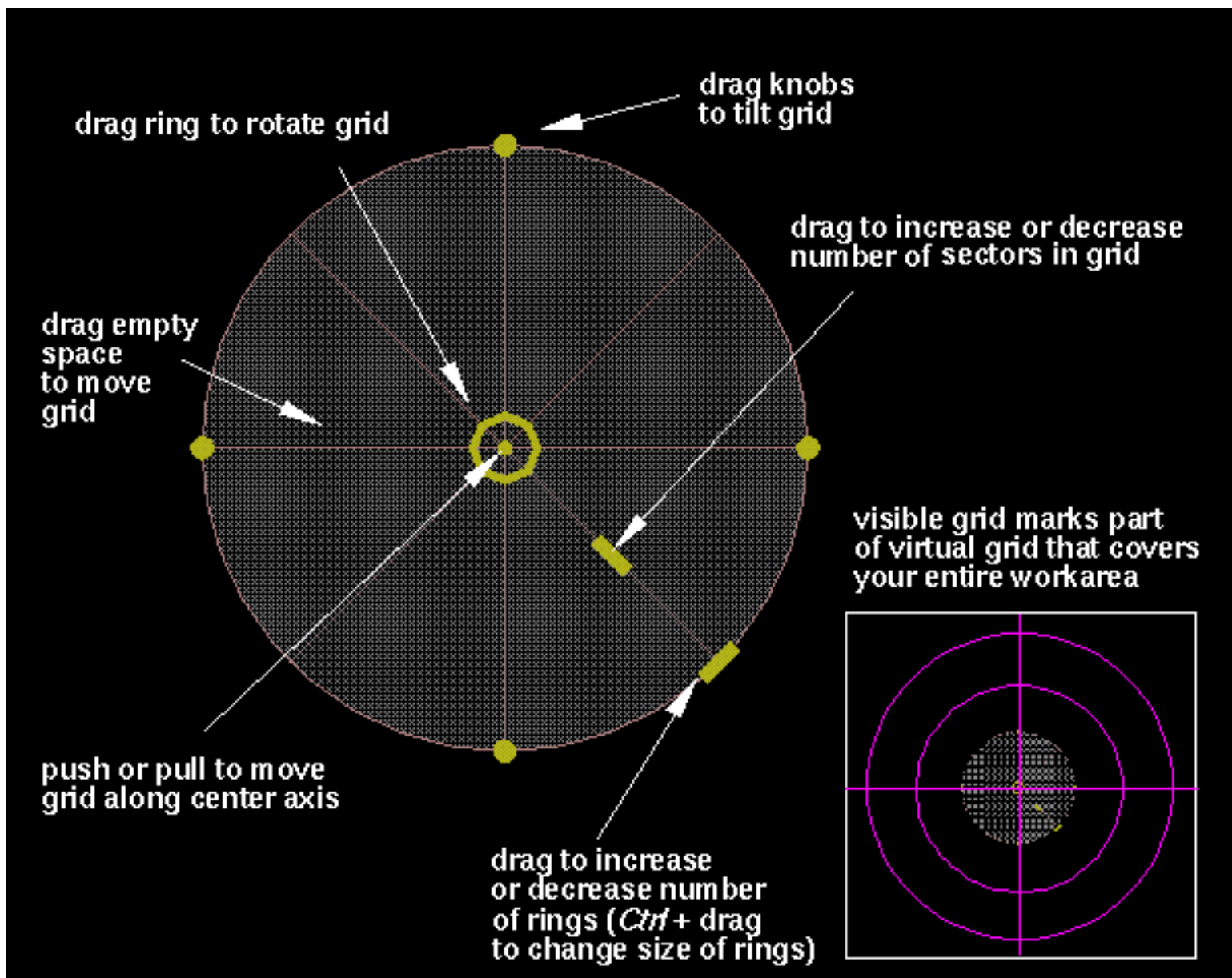
- The upright stick in the middle translates perpendicular to the plane.
- The round knobs rotate the grid out of its plane (tilting the grid). Both grids constrain the tilting to preset angle increments; press *Ctrl* to unconstrain and tilt the grid in a smooth motion.
- The ring in the middle rotates in the plane. This is also constrained to fixed increments unless you press *Ctrl*.
- The rectangular handle is for scaling. The Grid tool scales by increasing the number of divisions (if *Ctrl* is not pressed) or by the division size (if *Ctrl* is pressed). The Angle tool has two scale handles: one for rings and one for sectors. The ring handle increases the number of rings if the *Ctrl* key is pressed and the size of rings if it isn't. Since the number and size of sectors are complementary, the sector handle changes both. If you press *Ctrl*, the change is not constrained to increments of the current increment angle.

Double-clicking either tool brings up a panel for direct type-in editing of the tool size and layout.

Jump to:

- [Manipulating Models](#)
- [Snapping to Align Objects](#)





Snapping to Align Objects

on this page: [task summary](#) | [activating and placing](#) | [tips](#)

There are two kinds of targets you can use to align objects. The Snap Target (yellow arrow), and the Snap Source (green arrow). Use the Snap Target to mark a position in the scene to which you want another object moved or aligned. Use the Snap Source to specify a particular point on the selection to move to the Snap Target. The snap arrows can also be used in the PEP editor to align individual points, edges, and polygons. See [Aligning PEPs](#) for details.

Uses for the Snap Target:

- Aligning two objects. See [Task Summary](#)↗.
- Aligning an object to a grid. See [Layout Tools](#).
- Quickly placing a cut or copied object. See [Tips and Tricks](#)↗.

Use for the Snap Source:

- Specifying a part on an object to align with the Snap Target. For an example, see [Importing and Placing the Pontoon](#)s in the Rocket tutorial.

Task Summary

1. [Activate and place the Snap Target and the Snap Source](#)↗ (if you decide to use it).
2. Select the object you want to move to the target (click the object). Don't forget this step! If you don't select the object, it won't move to the target. If the Snap Source is activated, it will move to the Snap Target without the object.
3. Choose *Layout > Move Selection to Snap Target* or press *Ctrl-m* to move the selected object to the target. You can also choose one of the other options under Layout; these include *Center Snap Target in Selection* and *Move Selection Center to Snap Target*.
4. If you want, put away the Snap Target and Snap Source by rechoosing *Layout > Activate Snap Target / Source* or click the middle mouse button over empty space to put away the Snap Target, and *Ctrl-click* middle over empty space to put away the Snap Source.

Activating and Placing

To activate and place the Snap Target and Snap Source:

1. Choose *Layout > Activate Snap Target* or *Layout > Activate Snap Source*.

Other ways to place the Snap Target include:

- Click middle over an object's surface or over its manipulator.
- Click the *Activate Snap Target* button on the Layout palette:

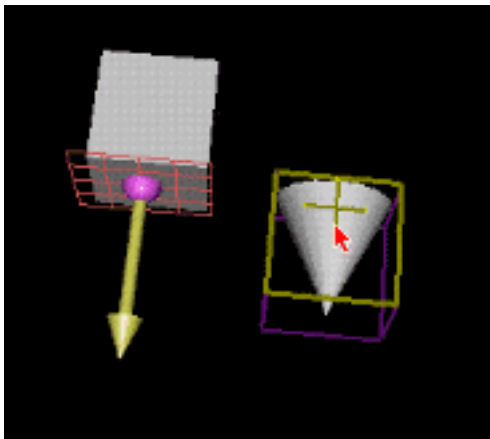


Other ways to place the Snap Source include: *Ctrl*-click middle over an object's surface or over its manipulator.

2. If necessary, reposition the Snap Target or Snap Source by dragging its arrow. If you have trouble getting the Snap Target to follow the cursor, try dragging while pressing the middle mouse button. This ensures that the target remains at the cursor and makes repositioning the target easier.

Tips

- *Shift*-drag makes the Snap Target "stick" to the object's major features. For example, if you *Shift*-drag the target across a cylinder, it sticks to the cylinder's edges and equator. If you *Shift*-drag the target across a PEP object, it sticks to the PEP object's vertices, edges, and edge centers. The target turns purple when it is aligned with one of these features.
- You can place the Snap Target to align an object's feature to the same plane as that of another. For example, suppose you have cube and a cone in your scene, and you'd like the cone's bottom polygon to align in the same plane as that of the cube. Place the Snap Target on one of the cube's polygons, then select and move the cone toward the cube until the Snap Target turns purple:



- If you cut an object and paste it when the Snap Target is in the scene, the cut object is pasted on the Snap Target.
- You can also nudge a selected object by small increments to adjust its alignment. Use the arrow keys or choose *Layout > Nudge > Left a Little, etc.*

Jump to:

- [Precision Placement Panel](#) (lets you specify coordinates for movement)
- [Layout Tools](#)
- [Manipulating Models](#)

Aligning Points, Edges, and Polygons

on this page: [snapping](#) | [local grid tools](#) | [setting options](#) | [global grid tools](#)

The PEP Editor features layout tools for aligning selected points, edges, and polygons. These tools include a snap target, a protractor, and a ruler. You can set preferences for the amount of "snap" these tools exhibit.

Use the protractor and ruler to move PEPs by increments relative to the PEP object's location. The [Grid and Angle layout tools](#) are also available in the PEP editor. Use these layout tools to move PEPs relative to the grid's location.

Set up:

1. Make sure you are in the [PEP editor](#).
2. [Select PEPs](#) that you want to align using the layout tools.
3. [Define the master selection](#). *Shift*-click to deselect and reselect correct PEPs if necessary.

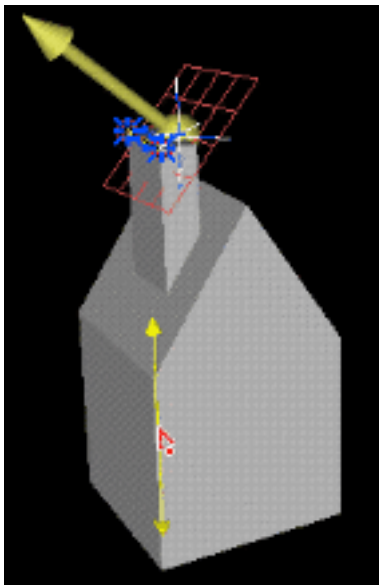
Snapping Selected PEPs

You can use the Snap Target to align PEPs when moving, rotating, or scaling them. Examples and How To's follow.

To move selected PEPs and snap to the target:

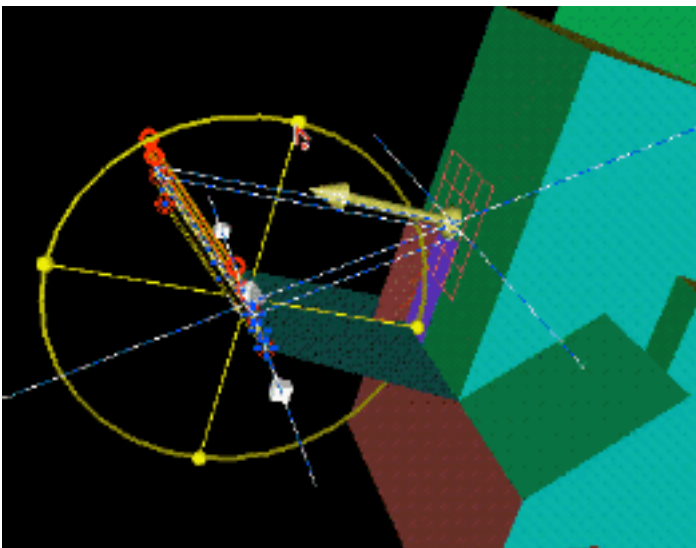
1. Place the snap target in the scene by choosing *Layout > Activate Snap Target* and dragging the snap target into position.
2. Move the selected PEPs to the desired location. The PEPs snap to the plane defined by the target.

For example, the chimney below was created by cutting a new polygon in the side of the roof, extruding the new polygon, then pushing the polygon up. However, this creates an angled chimney, because the roof is angled. To align the lower edge you would first select it, place the snap target (*Shift*-drag it until it turns purple, indicating that it is aligned exactly on the higher edge), then *Shift*-drag until the edge snaps to the plane. Three starbursts appear on the edge to indicate this alignment.



To rotate selected PEPs and snap to the target:

1. Place the snap target in the scene by choosing *Layout > Activate Snap Target* and dragging the snap target into position. Notice that *Shift*-dragging the snap target "sticks" it to a PEP object's points, edges, and polygons as you drag the target across these features. The snap target turns purple when it is placed at one of these features.
2. Place the [PEP Jack](#). PEPs rotate around the PEP Jack's center. So, if you want a door to open and close as if on a hinge, you would place the PEP Jack on the edge that would have hinges.
3. When you rotate PEPs using the Jack, notice that alignment feedback appears in the scene. This feedback consists of highlighted lines, as shown in the following example:



Also, the round balls on the PEP Jack become highlighted when the Jack's axis snaps to alignment.

Aligning PEPs with the Protractor and Grid

The Protractor and Grid help you align PEPs in increments local to the PEP object. For example, you

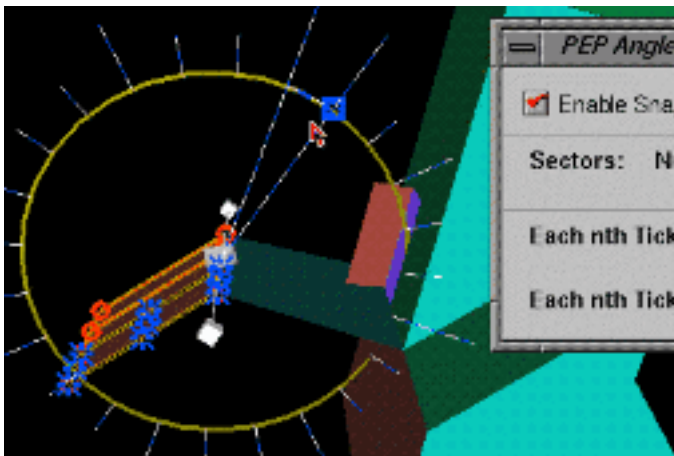
could rotate a model of a garage door 40 degrees from its previous position.

To rotate PEPs and snap to the protractor:

1. Open *PEP Angle Snap Options* by clicking its button on the *Layout* palette:



2. Click *Enable Snapping To Protractor*. Notice that you can specify the size and units of the protractor in this window.
3. Place the [PEP Jack](#) in the scene. When you rotate, bright tick-marks appear to indicate the Protractor. The rotation will snap to these ticks.



Notes:

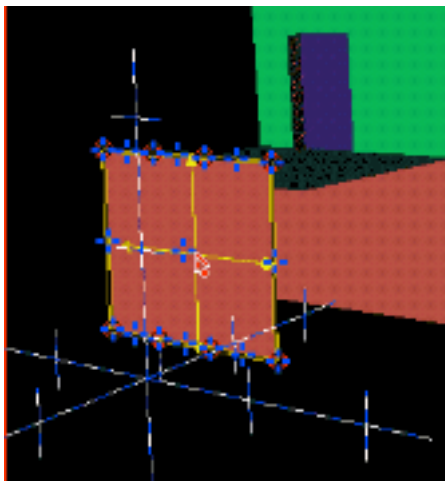
- Snapping to the Protractor occurs when:
 - you are in PEP editing mode
 - and you turn on *Enable Snapping To Protractor* in the *PEP Angle Snap Options* window
 - and you are using the PEP Jack to rotate selected points, edges, and polygons.
- **Exception:** If you have protractor snapping enabled and you have activated the Snap Target, the Snap Target overrides the Protractor.
- Use *PEP Angle Snap Options* not only to enable protractor snapping, but also to set the angular increments of the Protractor. (For details, see [To set protractor preferences.](#))
- *Shift* and *Ctrl* have no effect on protractor snapping.
- Features of the PEP Jack (the green balls, for example) are highlighted in blue when they snap to the Protractor's tick-marks.

To translate PEPs and snap to the ruler:

1. Open *PEP Distance Snap Options* by clicking its button on the *Layout* palette:



2. Click *Enable Snapping To Grid*. Notice that you can specify the size and units of the grid in this window.
3. Drag selected points, edges, and polygons. As you move the PEPs, they snap to the tick-marks on a light-blue grid:



Notes:

- Snapping to the grid occurs in the PEP editor when you have turned on *Enable Snapping To Grid* in the *PEP Distance Snapping Options* window. Simply drag selected PEPs to use the grid once it has been enabled.
- **Exception:** If you have grid snapping enabled and you have activated the Snap Target, the Snap Target overrides the Grid.
- Use *PEP Distance Snap Options* not only to enable grid snapping, but also to set the grid's increments and change its appearance. (For details, see [To set ruler preferences.](#))

Setting Options for Local Protractor and Ruler

To set protractor preferences:

- **Enable Snapping To Protractor:** Click the box to turn on protractor snapping. A checkmark appears to indicate that snapping is on. Click again to turn off.
- **Sectors:** Set the number and size of the protractor's sectors.
 - **Number** (default is 24 sectors)
 - **Size** (default is 15 sectors)
 - **Degrees, Radians, Revolutions** (default unit of measurement is Degrees)
- **Length of tick marks:** The protractor tick marks appear in 3 lengths. To increase readability,

you can change how often the lengths vary.

- **Each nth Tick drawn longer:** The default is for every 3rd tick mark to be drawn at a medium length.
- **Each nth Tick drawn longest:** The default is for every 6th tick mark to be drawn at the longest length.

To set ruler preferences:

- **Enable Snapping To Local Grid:** Click the box to turn on grid snapping. A checkmark appears to indicate that snapping is on. Click again to turn off.
- **X, Y, and Z Snap Increments:** Change the distance between tick marks on the X, Y, and Z axes. The default is 1.
- **Length of tick marks:** The ruler tick marks appear in 3 lengths. To increase readability, you can change how often the lengths vary.
 - **Each nth Tick drawn longer:** The default is for every 5th tick mark to be drawn at a medium length.
 - **Each nth Tick drawn longest:** The default is for every 10th tick mark to be drawn at the longest length.

Aligning PEPs Using the Grid and Angle Layout Tools

You can use the grid and angle layout tools in the PEP editor. See [Layout Tools](#) for details on using these tools.

PEP Jack

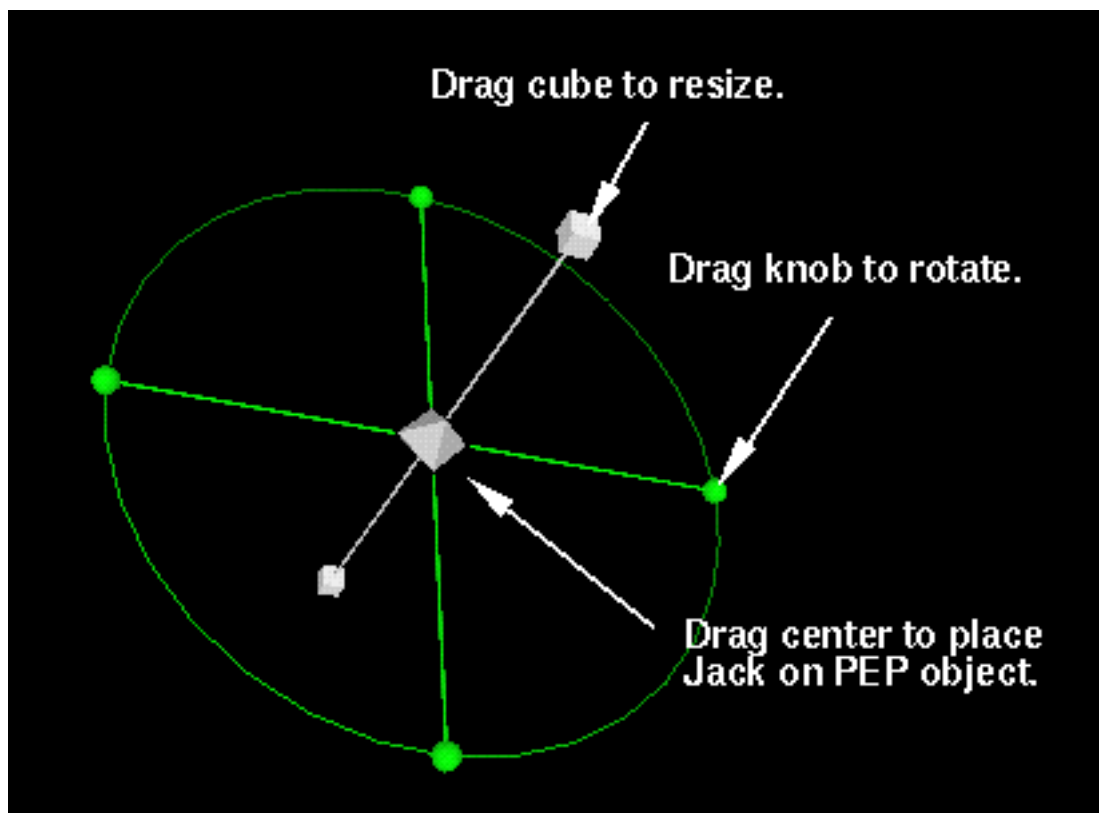
on this page: [how it works](#) | [examples](#)



Find it: Click its button on the *PEP* palette:

Use to: Rotate and scale points, edges, and polygons.

How It Works




You must be in the PEP Editor to use the PEP Jack. See [PEP Modeling: Editing Points, Edges, and Polygons](#) for details.

- **Positioning:** Drag the diamond in the center to translate the whole jack without moving the selected points.
 - Drag to move over the PEP object's surface. PEP Jack "sticks" to points, edges, and polygons of the PEP object.
 - *Shift*-drag will translate it in the plane of the ring, without re-aligning the axis.
 - *Ctrl*-drag will translate it linearly along the axis, without re-aligning the axis.
- **Scaling Points:** Drag the cubes at the end of the axis to scale selected PEPs about the center of the jack. *Shift*-drag constrains the scale to a single plane.

- **Rotating Points:** Drag the rotate knobs to rotate the selected PEPs about the center of the jack.
 - Drag for uniform scaling.
 - *Shift*-drag for free rotation about center
- **Snapping the PEP Jack:** If the Snap Target is enabled, then the PEP translation will snap to the target. For details, see [Aligning Points, Edges, and Polygons](#).

Examples

The following help pages show the PEP Jack in use:

- [Tutorial: Creating a Dormer for a House](#) 
- [Tutorial: Punching in the Back of the Ship](#)

Precision Placement Panel



Find it: Click its button on the *Layout* palette:

Use to: Type in values to scale, rotate, and translate.

How It Works

Coordinate System: Specify which coordinate system that the fields for Translate, Center, Rotate, and Scale use. *World* sets the coordinate system relative to the position of all objects in the scene. *Local* sets the coordinate system relative to the selected object's current position.

Transform: Use in conjunction with Coordinate System to specify the selected object's transform. *Relative* means to transform the object incrementally from its last position in World or Local space (whichever is specified). *Absolute* means to transform the selected object to the exact coordinates as specified in World or Local space (again, whichever is specified).

Translate To: Move object to location or by amount specified in Meters. Use the pull-down menu to choose a new measurement system.

Center; Move To: Set object's center of rotation, displayed in the scene by a selected object's green manipulator. Use the pull-down menu to choose a new measurement system.

Rotate By: Rotate by degrees specified in field. Use the pull-down menu to change degrees to radians or revolutions. (1 revolution = 360° ; 1 radian approx. = 57°) Choose to rotate around X, Y, or Z axes.

Scale To Size: Resize or stretch by typing in values for X, Y, and Z axes. Use the pull-down menu marked *Meters* to choose a new measurement system.

Apply: Click to apply the specified transform to the selected object in your scene.

Revert: Click to reset the fields to the last set of previously saved values.

Part 3: Animate the Rocket

In Part 3, you'll create an animation to make the rocket land on the planet.


Part 3 contains the following sections:

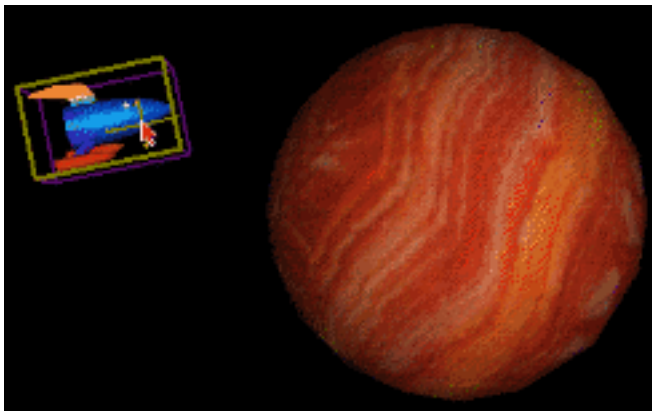
- [Setting Up](#) ↗
 - [Using the Keyframe Animator](#) ↗
-

Setting Up

1. If Cosmo Worlds is not open, open it now.
2. Choose *File > New*.
3. Import *shipLands.wrl*, a file provided for this tutorial. Choose *File > Import*. Clear the *Selection* field, then type in this pathname and press *Enter*: **/usr/share/Insight/library/SGI_bookshelves/Help/books/CosmoWorlds_UG/Models/shipLands.wrl**

Using the Keyframe Animator

1. Click on the rocket ship to select it.
2. Open the Keyframe Animator by pressing its button on the *Actions* palette:

3. From the Keyframe Animator's menu, choose *Animation > New Animation*. This creates a new, empty animation.
4. Now choose *Animation > Add Member* to add the rocket to the members of the animation cast.
5. Drag the large, right-pointed triangle (the time arrow) from the "0" mark to the "10" mark.
6. Move the rocket ship in your scene toward the planet.

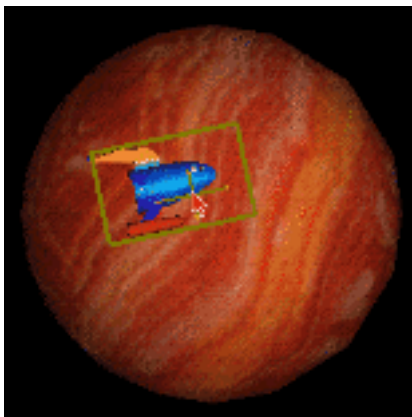


7. Notice that next to *Master* is a button that turned red:



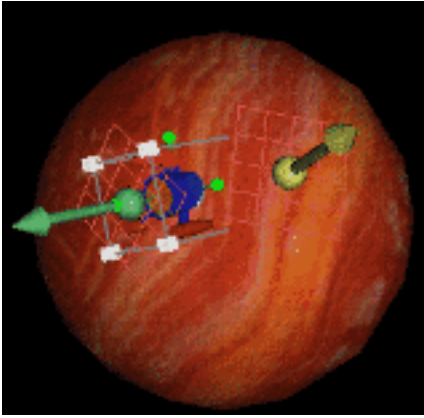
Click on this red button to record a keyframe at that time for all changed items in the animation. You have now set up a trajectory for the rocket ship to travel from time "0" to time "10."

8. Move the **time arrow** further to the right to the "15" mark.
9. Move the rocket ship closer to the planet.



10. Click on the red button next to *Master*.
11. Slide the timing arrow to the right from "15" to "20."
12. Click with the middle mouse button on the planet to select a landing spot. A yellow arrow appears. This is the Snap Target. To move the Snap Target, drag it across the planet's surface by holding down the middle mouse button. If you press *Shift* while dragging, the Snap Target will stick to the sphere's surface and turn purple when it is positioned on edges and other surface features.
13. Make sure the rocket is still selected. Move your mouse over one of the green knobs on the rocket's manipulator; it will turn orange. Hold down the left mouse button and drag to rotate the rocket so you can see its tail. *Ctrl*-click the middle mouse button in the middle of the

rocket's tail to place the Snap Source. A green arrow appears.



14. From the main menu (not that of the Keyframe Animator), choose *Layout > Move Selection to Snapping Target*. Or use the shortcut *Ctrl-m* instead. The rocket turns and moves to the Snap Target. This will be the final movement for your animation.
15. Set the duration to "20" for the animation. Choose *Animation > Set Duration* and specify 20.
16. Press the play button on the Keyframe Animator to see your animation. Drag the time arrow back to "0" before saving or previewing; starting the animation at its end could be confusing when you open the file again.

Jump to: [Keyframe Animator](#) for more information on this tool.

Next, you'll preview the animation in a VRML browser.


[Part 4: Previewing the World](#) ➡

Tutorial Part 4: Previewing the World

Now that you've finished the content of your world, take a look at it using Cosmo Player. You'll test it to see if it looks and behaves as it should.


1. Make sure Cosmo Player and Netscape are installed on your system.
2. You should have *shipLands.wrl* still open from Part 3 with the animation you created.
3. In the Cosmo Worlds main menu, choose *File > Preview*. This launches Netscape and displays your world with the Cosmo Player plug-in. You may need to reposition your view of the planet if your rocket is hiding behind it. To learn how to define viewpoints for the browser, see [Defining Viewpoints](#).
4. Click on the rocket to see if it flies to the planet and lands on it.

Congratulations! You've now learned how to create shapes, edit them using the PEP modeler, change their colors and textures, and add animation to your scene. Next you could:

- Learn more about Cosmo Worlds in the [Overview](#) or [Getting Started](#).
- See some example [VRML worlds](#). 
- Experiment on your own!

Getting Started

Cosmo Worlds is all about creating a 3D world that can be published on a Web server. This includes making your world efficient for downloading and viewing.

The following topics assume you are familiar with [VRML concepts](#). The topics appear in the order that you might complete each task when you create and publish a world for the first time.

1. Learn [how to view objects and worlds](#).
2. Learn [how to manipulate objects](#).
3. [Create basic shapes and text](#) which you can [model into more complex shapes](#). If the models are complex, make a separate file for each model, then import the models into a world.
4. [Import](#) and edit models created with other modeling software programs.
5. Add [colors](#), [textures](#), and [materials](#) to your models.
6. [Optimize the world](#). Reduce the number of polygons in your models. Create level-of-detail (LOD) groups where appropriate. Use inlines.
7. [Animate](#) models in your world and write your own [scripts](#) to define additional behaviors for the objects.
8. Create special objects, such as [lights](#), [sounds](#), movies, and [links](#) to other files.
9. Use the Outline Editor to edit the VRML file (a readable text file) for your world. The [Outline Editor](#) also allows you to create [routes between events](#) so that values can be sent from one part of the world to another part.
10. [Save](#) the world to a file.
11. Preview the world so you can test links, levels of detail, animations, sounds, and movies, as well as overall frame rate and performance.
12. [Package](#) the world and publish it on the World Wide Web.

Jump to: For more information on VRML, see *The VRML 2.0 Handbook*, by Jed Hartman and Josie Wernecke or jump to the [VRML 2.0 specification](#).

Creating Basic Shapes and Text

Creators let you create and place basic shapes and text. Once you create and place a shape, you can further [manipulate the object](#) or edit it in the PEP Editor; see [PEP Modeling: Editing Points, Edges, and Polygons](#) for details.

Find it: Click a button from the *Create* palette:



The **cube**, **sphere**, **cone**, and **cylinder** buttons create those shapes. After you click a button, [interactively place](#) the shape in your work area.



The **Draw Face** button lets you draw a 2D shape. You can draw on top of existing objects. See [Drawing a Face](#) for more details.



The **Extrusion Editor** button lets you create and [interactively place](#) a default shape that you can extrude using the [Extrusion Editor](#), which launches when you place the shape.



The **Text** button lets you create and [interactively place](#) default text that you can edit in the [Text Editor](#), which launches when you place the default text.

Placing Shapes When Importing, Pasting, and Creating

on this page: [importing](#) | [pasting](#) | [creating](#) | [interactive placement](#)

You can interactively place shapes on import, paste, and creation. This means that the object first appears "stuck" to the cursor. Move the cursor over your work area and click to place the shape. Drag your cursor to resize the shape and let go to place it. Details on [interactive placement](#) appear below.

You can instead [place an imported](#) or [pasted](#) object in its original coordinates by turning off interactive placement.

Interactive placement is the only way to place a basic shape when you click one of the creator buttons on the Create palette.

Placing an Imported Object

How you place a shape on [import](#) depends on whether you have *Interactive Import Placement* turned on. By default, it is turned on the first time you launch Cosmo Worlds.

Find it: Choose *File > Interactive Import Placement*

Deselect *Interactive Import Placement* to place the object in its original coordinates; that is, the location the object last appeared in its native environment before you imported it.

Pasting a Cut or Copied Object

How you paste an object that you have cut or copied depends on whether you have *Interactive Paste Placement* turned on. By default, it is turned on the first time you launch Cosmo Worlds.

Find it: Choose *Edit > Interactive Paste Placement*

Deselect *Interactive Paste Placement* to place the shape in its original coordinates.

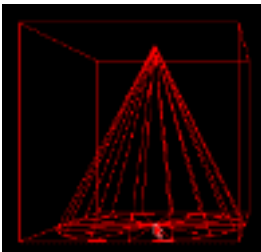
Placing a Basic Shape

Shape creators let you interactively place, size, and scale basic shapes. Click on a button and interactively place in your work area.

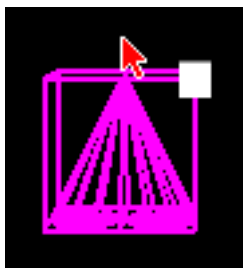


Interactive Placement

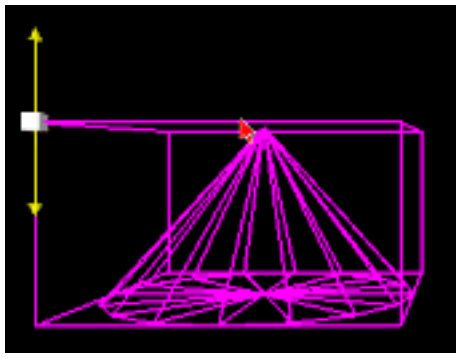
1. When you import, paste, or create a shape, a wireframe version of the object appears under the cursor.



2. **Move your mouse in the scene, but don't click yet.**
3. The object will:
 - Snap to a snap target if one is in the scene (object axes match snap target axes).
 - Follow surfaces.
 - Appear midway between front and back [clipping planes](#). The object aligns to world space axes. The direction of your object matches the world axis closest to "up" in screen space.
4. **Click to create the object at the size you see.**
5. **Drag-release to size an object before placing it.**



- **Drag-release to scale proportionally.** Drag out from the starting point. The direction of your gesture chooses from four possible dragging lines that match the diagonals of the bounding cube. As you drag along the line, you'll make the object bigger/smaller.



- **Shift-drag-release to stretch.** Press the *Shift* key and three orange arrows appear. Your next gesture picks from among the three directions for non-proportional scaling. When you move far enough, the chosen direction is displayed with a single yellow arrow.

6. Other Resizing Key Combinations

- **Sweeping Out Corner-to-Corner (*Ctrl-drag*)**--By default, the object is swept from the bottom-center point out to the corner. So the initial click denotes the center of the object's base. If, instead, you want to sweep out from one corner of the bounding cube to the opposite corner, hold *Ctrl* when you drag out.
- **Switching Back and Forth**--You can switch back and forth between scale and stretch by pressing/releasing the *Shift* or *Ctrl* key as you drag.
- **Negative Scales**--The creators will come up with equivalent transforms to put the object in the same place but without any negative scaling.

Drawing a Face

Use the Draw Face button from the Create palette to draw a 2D face:



You can draw on top of an existing object by clicking the Draw Face button, then click on the object to begin drawing.

The Draw Face tool is a bezier pen. Click to create an initial point, then move the cursor to position a second point and click to create a straight line. Dragging creates a center point around which you can bend the line between two points.

Click-middle to close-off and finish the shape.

Jump to: [Creating Basic Shapes and Text](#)

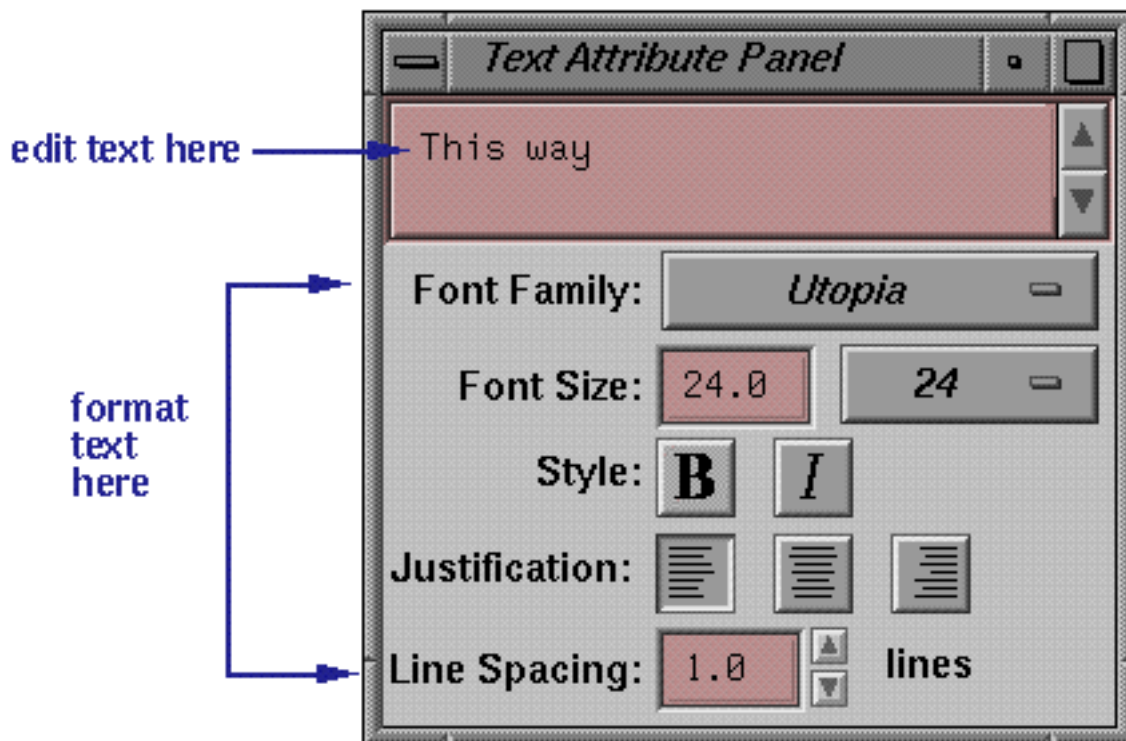
Text Editor



Find it: On the *Creators* palette, click VRML text

Use to: Edit VRML text in your scene.

How it works:



Jump to:

- [Placing and Editing Text](#)
- [Applying a Texture](#)
- [Changing the Material](#)
- [Adding Color](#)

Placing and Editing Text

You can place VRML text in your scene.



Find it: Click its button on the Creators palette:

1. To create text, click the VRML text button, and place the text in your scene. The text temporarily says "3D Text."
2. Use the [Text Editor](#) to type in a new word or sentence. You can also change the font size and attributes.
3. Once you place and edit the text, you can [change the color](#), [add a material](#), or [add a texture](#) to the text.

Adding Color to Models

The Color Per Vertex editor lets you add color to models which will be converted automatically to PEP objects if needed. You can also edit color that you have already added to a PEP object.

To learn how to add color to a model, follow the basic steps below. For more details, see [Color Per Vertex Editor](#).

1. **Select the object** you want to color (click on it).
2. **Open the Color Per Vertex editor** by clicking its button on the *Looks* palette:



3. **Reposition the object.**
 - *Alt* turns the cursor into a hand icon. Drag the object to **reposition**.
 - *Alt*-click-drag and let go to **spin** the object. Click to stop spinning.
 - *Ctrl-Alt*-drag middle up or down to "**zoom**," or dolly in or out.

4. **Choose how you want the color to be applied.**



Drag your cursor over each button for quick help--a description of the button appears in the editor window. The icons describe how the color will appear on the selected vertex:

- The first two icons produce a fade effect. The first icon changes the color of a single vertex. The second icon changes the color of adjoining vertices.
- The third icon changes the color of the entire face. This results in solid color.
- The fourth icon lets you quickly edit all faces with that same color.
- The fifth icon colors the whole object.



5. Click the arrow and **choose a color** from the [color editor](#).



6. Choose the paint brush, and drag to apply color.

Tip: You can quickly switch from choosing a color (arrow) to applying a color (paintbrush) by pressing *Ctrl*.

7. Click *Apply* when done.


Jump to:

- [Changing the Material on an Object](#)

- [Applying a Texture](#)
- [PEP Modeling: Editing Points, Edges, and Polygons](#)

Changing the Material of an Object

Use the Material Palette to replace the material of an object.

1. Select the object.
2. Open the [Material Palette](#) by pressing its button in the *Looks* palette:
A small, square button with a grid of colored circles, representing the Material Palette button in the Looks palette.
3. Select a palette or [create a new one using the Material Editor](#).
4. Click on a sphere in the palette to apply its material to your selected object.

Jump to: [Creating & Editing Materials](#)

Creating New Material Palettes

Find it: *Looks* palette > *Material Palette*



1. Open the [Material Palette](#).
2. Choose *File* > *New*. A dialog box appears.
3. Enter a name for the palette you're creating and click *OK*. A new palette appears. The gray spheres don't have materials yet. You'll need to create them.
4. Double-click a sphere to open the [Material Editor](#) (or choose *Edit* > *Edit Material*).
5. Use the Material Editor to [create and edit new materials](#).

Jump to:

- [Tools to Use: Creating & Editing Models](#)
- [Applying Color to Models](#)

Tools to Use: Creating & Editing Models

on this page: [starting out](#) | [chipping off and editing faces](#) | [extruding](#) | [copying and pasting](#) | [repairing and reducing](#) | [coloring and adding texture](#)

Starting Out

Find it: *Tools > Content Editor Tool Palette (Editors), Appearance Tool Palette (Looks)*

[PEP Creator](#)

Select VRML object or other imported shape, click PEP creator button.



The object becomes a PEP object, and can now be edited using Cosmo Worlds's PEP editing tools (listed here).

[PEP Editor](#)

Select a PEP object and press *Ctrl-e* or click the PEP editor button:



A red box appears around the object while in PEP editing mode.

[PEP Selection](#)

After clicking PEP Editor button, you can click on a point, edge, face, or any combination of the three for editing. *Click and drag* to select a group with a marquee. *Shift* and *click* to select multiple points, edges, faces by selecting them individually.

Chipping Off and Editing Faces

[Chip off](#)

Select face to "chip off." Click *Chip Off Selected Polygons* button.



Pull selected face away from object.

[Divide Faces](#)



Select a polygon face, click a button to split face into smaller triangles or quads. You can now edit the new angles (extrude, chip-off, paste to another PEP object, etc.).

Extruding Polygons and Edges

Extrude Polygon

Select polygon. Click extrude button.



[Move selection](#) by dragging.

Extrude Edges

Select edges. Click Extrude Edges button:



[Move selection](#) by dragging.

Copying and Pasting Faces

Choose from a variety of functions for cutting, copying, and pasting. See [Merging, Combining, or Joining PEPs](#) for details.

Repairing and Reducing

Polygon Copier



Use to rebuild a PEP object to create a more efficient model (one with fewer Polygons). This one has its own interface.

Normal Doctor



Use after editing a PEP object to specify render style, back-face culling, and polygon ordering.

Polygon Reducer



After using Polygon Copier and Normal Doctor (if necessary), use the Polygon Reducer to finalize the object's number of polygons.

PEP Jack

Use to rotate and scale selected points, edges, and polygons. Create joints for rotation (e.g. doors).

To use, select points, edges, faces to be rotated. Press the Rotating/Scaling Points button.



Grab tool at center and drag to place (snaps to object's points as you drag it). Drag green knobs to twist/rotate points, or drag white cubes to resize points.

To put away tool, press its button again.

Coloring and Adding Texture

[Color-per-Vertex Editor](#) 

Apply color to polygon. Paint color on with the editor.

[Texture Editor](#) / [Material Palette](#)

Apply texture, material to object. A more effecient way to spruce up a model is to add color.

Buttons are Texture Editor , Material Editor , Material Palette 


Chipping-off and Forming New Polygons


on this page: [how to](#) | [example 1](#) | [example 2](#)

How To

Set up:


1. Make sure you are in the [PEP editor](#).
2. [Select polygons](#) that you want to delete.

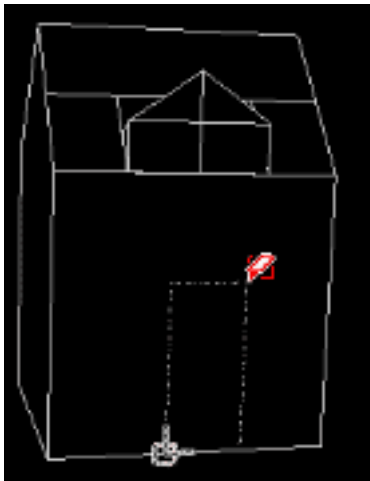
To chip a polygon off a PEP object: press 
Now you can pull away and reposition the face.

To form a new polygon from selected edges: press 
A new polygon forms over the selected edges.

You can also extract polygons into new PEP objects. For details, jump to: [Merging, Combining, and Joining Polygons](#).

Example 1: Use Chip Off Polygon to Create a Door

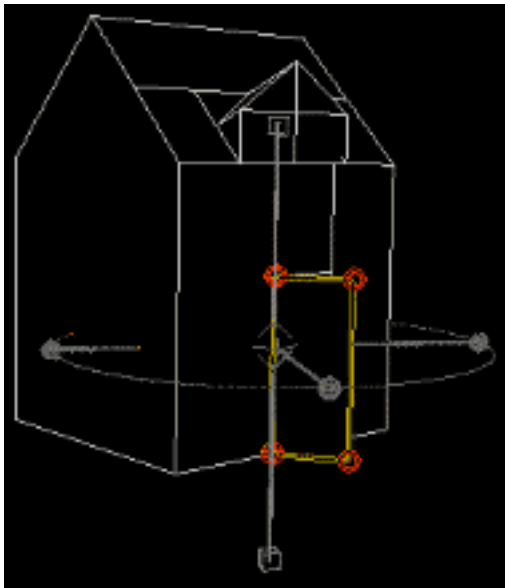
Suppose you want to create a door for a house. First, you'd use the cut rectangle tool  to define a new polygon for the door:



But if you try to select and move this polygon, it will pull out the points that define the front of the house. To move just the polygon that defines the door, select it and click the chip off polygon button:



The result will be a polygon you can move independent from the rest of the house. Next, you probably want to rotate the edge of the door like a hinge:



For this you'd use the [PEP Jack](#), a tool for rotating and scaling points and edges.

Example 2: Use Form New Polygon From Selected Edges With the Draw Tool



The Form New Polygon From Selected Edges  is very useful when you use the draw tool to create unique shapes.

Suppose you want to make a star shape to use in your scene.




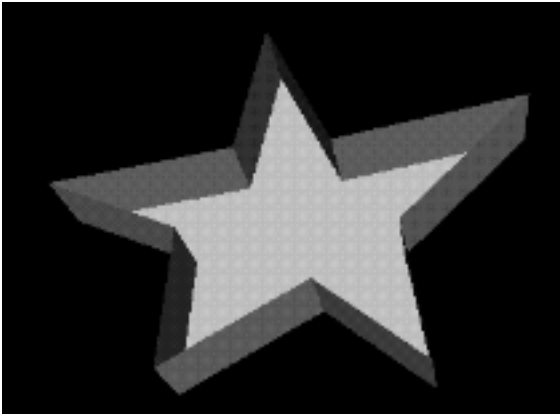
1. Click  on the Create palette and click-drag to create a star shape. **Jump to:** [Drawing a Face](#) for more information.



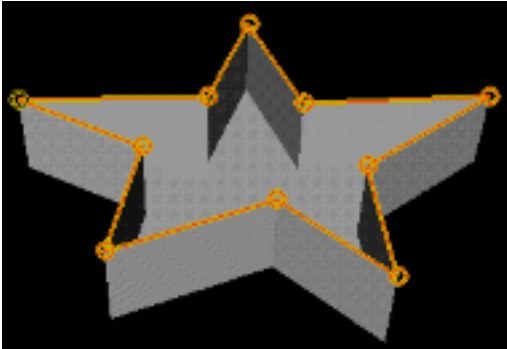
2. Select the shape and enter the PEP editor by clicking .




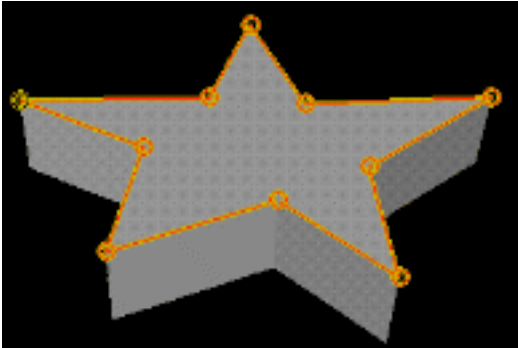
3. Click Extrude Polygons  and *Ctrl*-drag to pull the selected polygons up. Now you have a nice extrusion, but the bottom cap is missing (reposition your view so you can see the bottom):



4. [Sweep select](#) the bottom edges:



5. Click  to form a new polygon over the selected edges:



Jump to:

- [PEP Modeling](#)
- [Tools to Use: Creating and Editing Models](#)

Merging, Combining, or Joining PEPs

on this page: [how to](#) | [example](#)

How To Merge, Join, or Extract Polygons

Set up:

1. Make sure you are in the [PEP editor](#).
2. [Select polygons](#) that you want to merge, combine, or join.

To merge selected polygons: press



This merges any number of adjacent, selected polygons into a single polygon on the same PEP object.

To join selected PEPs into a single PEP object: press



This takes separate, whole PEP objects and merges them into one single PEP object. For example, suppose you have modeled four wheels and a car body. Select all elements, the car body and the wheels, then click the button to create a single PEP object.

Note: This function is enabled only if you have multiple PEP objects selected. You are usually not in the PEP editor to use this function.

To extract selected polygons into a new PEP object: press



This takes the currently selected polygons (it does not include single edges or points that are not parts of selected polygons) and cuts them out of the currently selected PEP object. Then it makes a brand new PEP object and puts those polygons in the new, separate object.

You would use this, for example, if you cut a door out of a house and wanted to separate it into its own object so you could animate the door opening and closing using the rotation in its Transform.

To merge colinear edges in a selection: press



This merges all multiple, end-to-end edges (in the selection) into a single edge with two endpoints.

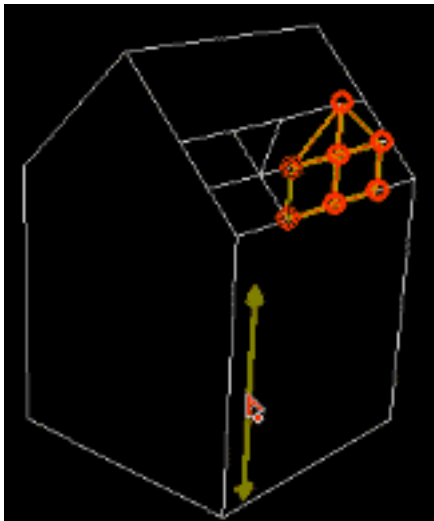


To combine selected points onto the master selection: press

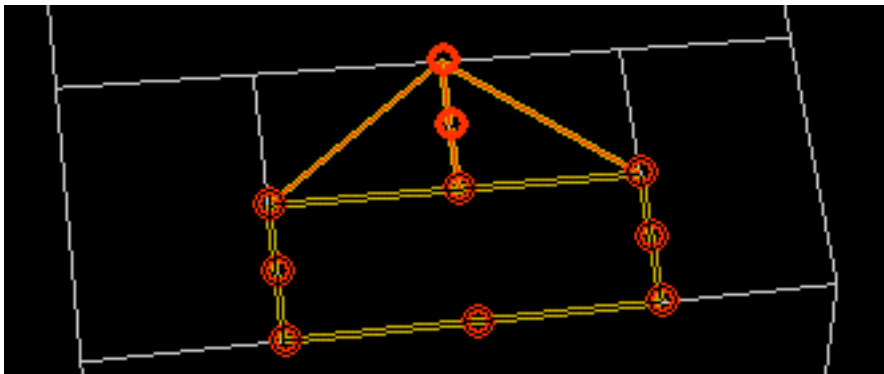
This combines all selected points into a single point, which is defined by the [master selection](#).

Example: Cleaning up Colinear Edges

Cutting and splitting polygons often creates extra edges and points in the model. For example, suppose you want to create a house with a dormer as shown in the following image:



You've created the dormer in an inefficient way that will still work, but produces colinear edges:



You can solve this problem by [merging colinear edges in the selection](#).

Jump to:

- [PEP Modeling](#)
- [Tools to Use: Creating and Editing Models](#)

Splitting and Cutting Polygons

on this page: [how to](#) | [try it](#)

How To

Set up:

1. Make sure you are in the [PEP editor](#).
2. [Select polygons](#) that you want to split or cut.


To split a polygon using pre-defined operations:

1. Press the appropriate button from the *Split* palette:




2. Now you can select and edit the new faces.

To split a polygon by dragging the cutting tool:

1. Press this button: 

2. Drag from one side of a polygon to another to split the polygon along a new edge.

To cut a rectangle out of a polygon:

1. Press this button: 

2. Drag out a rectangle. When you end the drag, the original polygon is split into several polygons.

See [Chipping-Off Polygons](#) for an example of using this tool to create a door in a house.

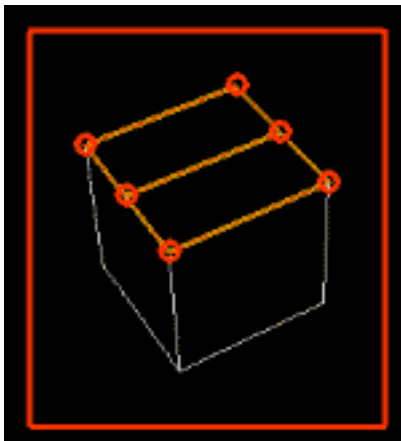
Try It: Creating a Roof for a House

The basic shape for a house is fairly simple.

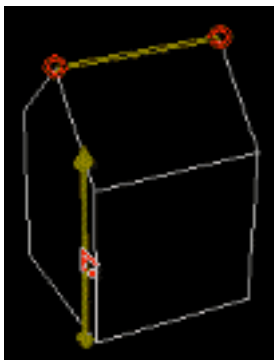
1. Start out with a cube in the PEP editor.



2. Select the top polygon and press  to split the top polygon lengthwise.

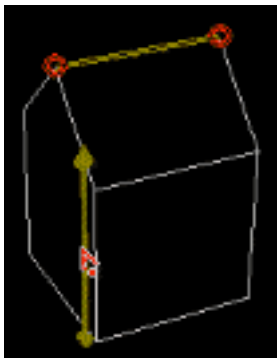


3. Select the edge in the middle of the split polygon and *Shift*-drag along a side polygon to push the edge up and form the roof:



Try It: Creating a Dormer for a House

You can model interesting shapes from a simple PEP object like a square by using a combination of split and cut operations, then moving and extruding the new polygons. For example, suppose you want to create a dormer for a house that looks like this:

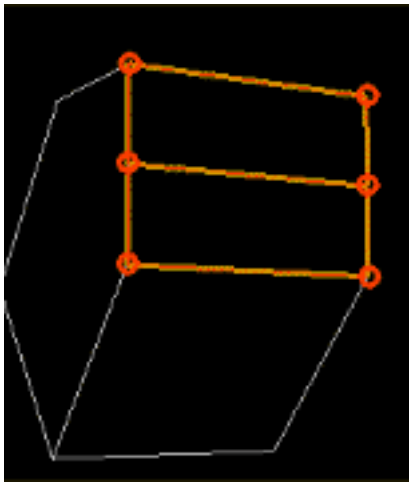



The combination of split buttons you would use follows:

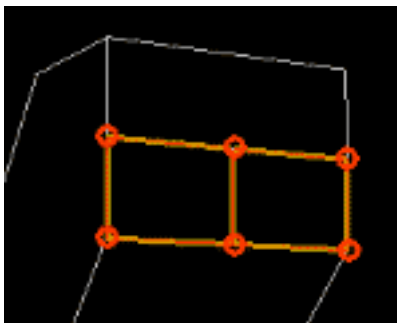



1. Starting with the basic house shape in Example 1, select a side of the roof and press  to

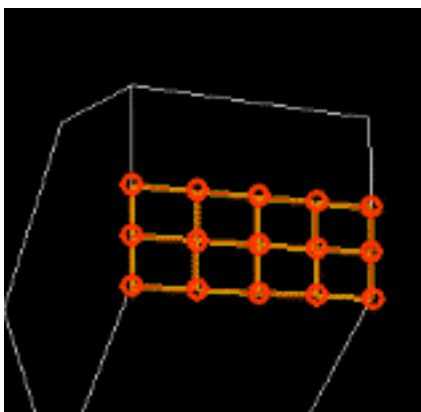
split the polygon lengthwise:





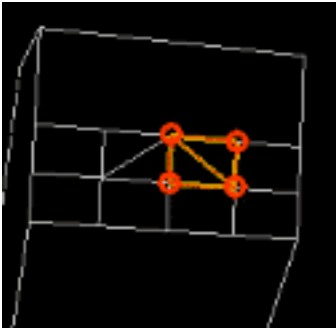
2. Select the lower half of the roof and press  to split the polygon vertically:




3. Keeping the same selection from step 2 (don't select anything new), press  to split the quads like this:



4. To make the angles for the dormer roof, select the appropriate quad and press  or .



Now you have the basic shape for the dormer. To pull out the dormer, you'd select the polygons that make up the dormer, press the Extrude Polygons button, and move the polygons out so the front of the dormer is flush with the front of the house. See [Extruding Polygons and Edges](#) and [Aligning PEPs](#) for more information on these topics.

To complete the tutorial, see [Creating a House](#) .

Jump to:

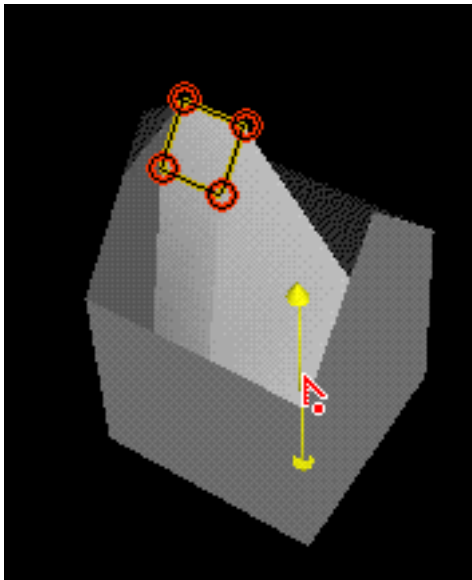
- [PEP Modeling](#)
- [Tools to Use: Creating and Editing Models](#)

Extruding Polygons and Edges

on this page: [moving vs. extruding](#) | [how to](#)

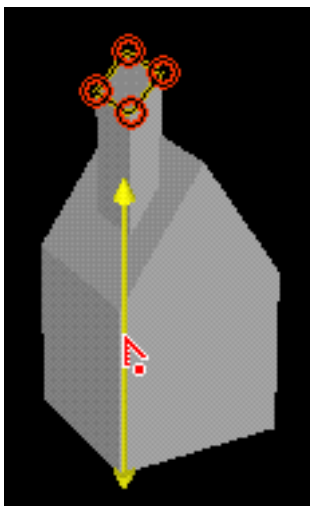
Moving vs. Extruding

When you move, or translate a selected polygon, the polygon moves to the position specified by dragging. When you extrude a selected polygon you are not moving it, but duplicating it so that when it is moved, new sides form connecting the polygons. So when you extrude a polygon and then move it, the surrounding polygons stay the same shape; compare this to how the surrounding polygons change shape if you do not extrude the polygon first. For example, if you create a house with a chimney and move the selected polygon before extruding it, here's what will happen:



Not quite the right thing.

But when you move the polygon after extruding it, a nice rectangle shape is formed:



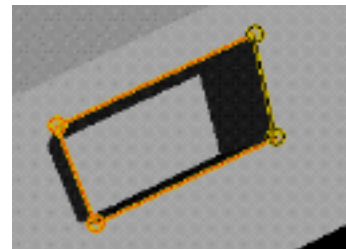
Just right for a chimney.



Note that this example uses Extrude Polygons



If you had used Extrude Edges instead, the chimney would have no top:



Trick: If you accidentally use Extrude Edges, you can close the top:

1. Make sure the correct edges are selected.

2. Click Form New Polygon From Selected Edges



How To Extrude Polygons or Edges

1. Make sure you are in the [PEP editor](#).

2. [Select polygons or edges](#) that you want to extrude.

3. Click the Extrude Polygons button:



or the Extrude Edges button:



4. [Move the extruded polygons or edges](#) to a new position.

Jump to:

- [PEP Modeling](#)
- [Tools to Use: Creating and Editing Models](#)

Texture Editor

on this page: [feature highlights](#) | [menu bar](#) | [options menu & thumbwheels](#) | [sliders](#)

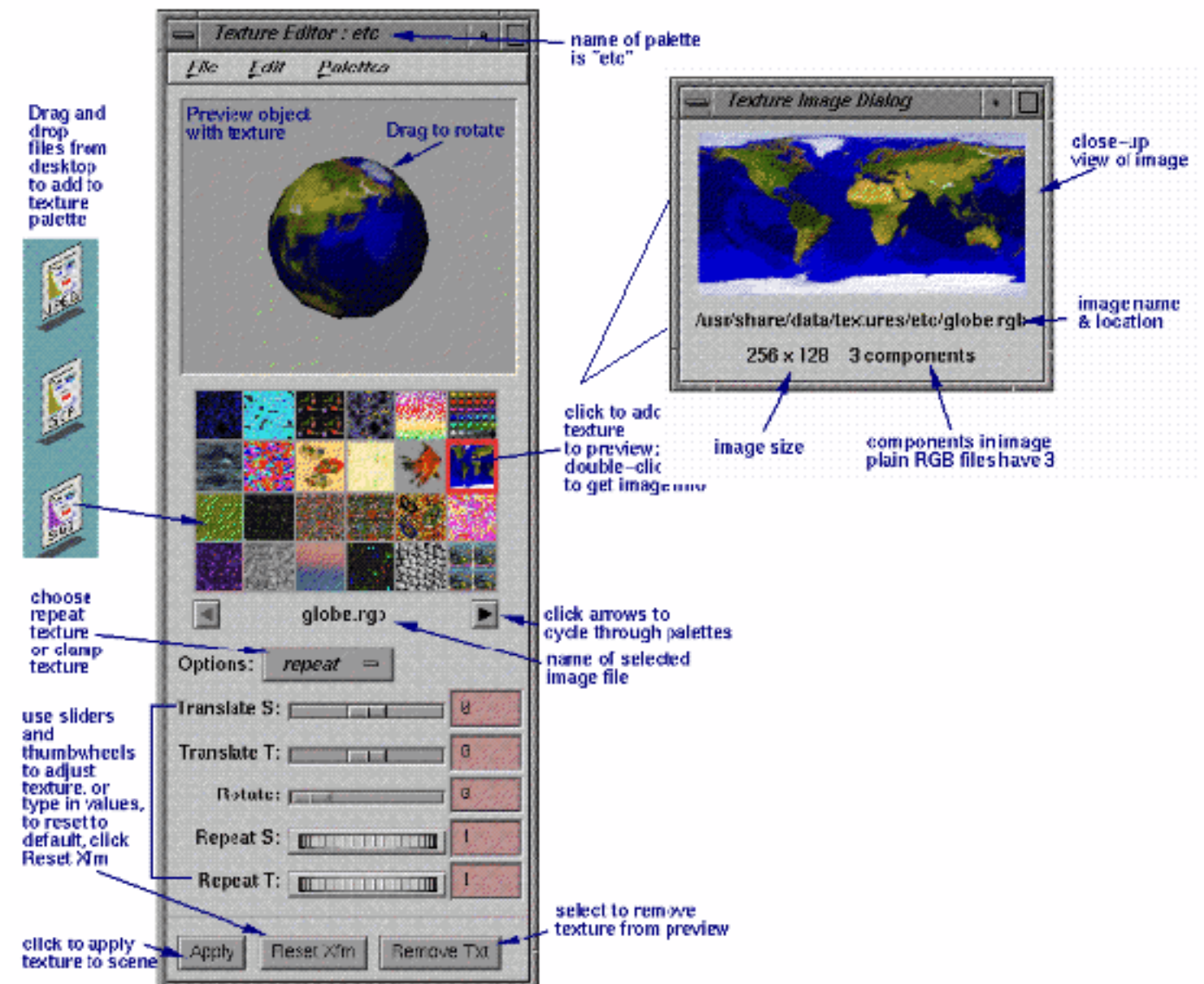
Find it: Click its button on the *Looks* palette:



Use to:

- [Apply a texture to an object](#)
- [Replace textures with new ones](#)
- [Create a custom palette of textures](#)

How it works:



Click [Texture Editor](#) and [Texture Image Dialog](#) for full view.

Feature Highlights:

- Drag and drop GIF, JPEG, PNG, and RGB (SGI-image) files onto the texture palette.
- Get information on each texture image by double-clicking it, or choose *Edit > Show Image*. This opens the Texture Image Dialog, which displays the image size, image name and location, and the components to the image. RGB files can consist of one to four components:
 - 1 = Black and White (BW)
 - 2 = BW plus Alpha Transparency
 - 3 = Red, Blue, Green (RGB)
 - 4 = RGB plus Alpha Transparency
- By default, the texture is mapped to the largest face of object. The texture is wrapped to cover smaller faces by stretching the image. Cube, Cylinder, Sphere, and Cone have default mappings.
 - To adjust the image, select *repeat* or *clamp* from the Options pull-down button and experiment with the sliders and thumbwheels.
 - For fine-tuning, use the [PEP Texture Applicator](#).

The Menu Bar:

- **File:**
 - *New* lets you [create a custom palette](#).
 - *Import Image* lets you import a file into an empty texture square.
 - *Warning On Size* warns you if the image you are importing is larger than 128x128 pixels. Large images will affect your scene's rendering performance, so try to keep texture images to a minimum size.
 - *Reset System Palette* lets you return to the default palette. Use this if you've modified a texture and want to revert to the original.
 - *Delete Custom Palette* lets you delete any custom palettes that you've created. You cannot delete the default palettes.
 - *Close* closes the Texture Editor.
- **Edit:**
 - *Show Image* shows a larger image with details on image size and type.
 - *Delete Image* removes a selected texture.
- **Palettes:**
 - The Palettes menu lists all the texture palettes installed on your system.
 - Go directly to a texture palette by selecting its name.
 - Use *Next* and *Previous* to page through the palettes.

The Options Menu & Thumbwheels:

- Use to specify scaling and repetition of a texture on an object:
 - Choose a *Repeat* menu item; thumbwheels read *Repeat S* and *Repeat T*
 - Choose *Clamp*; thumbwheels read *Scale S* and *Scale T*
- *Repeat* specifies the number of times you want the texture to appear on a model. Use the *Repeat S* and *Repeat T* thumbwheels or type-in fields to specify.
- *Clamp* makes the texture larger or smaller on the model without repeating it. The edge of the texture is smeared across the object instead. Use the thumbwheels labeled *Scale S* and *Scale T* to make the texture larger or smaller.
- *Repeat S* repeats the texture in one dimension of the object. Use the thumbwheel labeled *Repeat S* to display more or fewer repetitions.
- *Repeat T* repeats the texture in another dimension of the object. Use the thumbwheel labeled *Repeat T* to display more or fewer repetitions.

The Sliders:

- *Translate S* sets the position of the texture in one dimension on the model. The default is centered. Use the slider to move the texture sideways, or use the type-in field.
- *Translate T* sets the position of the texture in the other dimension on the object. The default is centered. Use the slider to move the texture up or down, or use the type-in field.
- *Rotate* specifies whether the texture is applied in an upright position or rotated. Use the slider to rotate the texture or use the type-in field.

Applying a Texture

1. Select an object in your scene.
2. Open the Texture Editor by clicking its button on the Looks palette:



3. Choose a texture palette from the *Palettes* menu.
4. Click a texture. It appears in the display area of the [Texture Editor](#) so you can preview it.
Click and drag to rotate the object in the display area. To choose a different texture, just click a new one. The new texture replaces the previous texture in the display area.
5. Click *Accept* when you want to apply the texture to the selected object in your scene.

Jump to:

- [Editing a Texture](#)
- [Creating New Texture Palettes](#)
- [Fine-tuning a Texture](#)
- [Texture Editor Reference](#)

Editing a Texture

To edit a texture, create a new GIF, JPEG, PNG, or RGB image. Delete the old texture from its palette. Drag-and-drop the new texture file into the palette.



Find it: *Looks* palette > *Texture Editor*

1. In the [Texture Editor](#), select the texture you'd like to change.
2. Delete the texture. Choose *Edit* > *Delete Image*.
3. Replace the empty space with a new texture. Choose *File* > *Import Image*, or drag-and-drop the image file from your desktop.

Click *Accept* when you want to apply the texture to the selected object in your scene.

Note: Dragging and dropping an image onto the palette will add it to the palette. Dropping it onto the model will apply it to the model (without adding it to the palette).

Jump to:

- [Applying a Texture](#)
- [Fine-tuning a Texture](#)
- [Creating a New Texture Palette](#)

Fine-tuning a Texture

Once you apply a texture to an object, you may want to adjust parts of the object by selecting specific polygons in the object and moving, rotating, or scaling the texture by manipulating those selected polygons.

Find it: *Looks palette > PEP Texture Applicator*



To make the PEP Texture Applicator button available (not grayed-out):

1. Select an object in your scene.

2. Make the object into a PEP model by pressing



3. Open the Texture Editor, and [apply a texture](#).

To use the PEP Texture Applicator:

1. Open the PEP Texture Applicator by pressing



2. In the PEP Texture Applicator, [select the polygons](#) on which you'd like to adjust the texture.

3. [Remap the texture](#).

4. Click *Accept* when you want to apply the changed texture to the selected object in your scene.

Jump to: [PEP Texture Applicator Reference](#)

Selecting Polygons in the PEP Texture Applicator

When you're fine-tuning a texture using the PEP Texture Applicator, you select the polygons to manipulate in the right side of the editor's window. Adjust your view here the same way you would in the Examiner Viewer. See [Viewer Controls and Menus](#) for details.

To select polygons:

1. Make sure the cursor is in the right side of the PEP Texture Applicator's window (the one with the 3D view).
2. Click any part of the object to select a polygon. Shift-click to select multiple polygons.

You can also drag the mouse to select multiple polygons, but note that this method selects all the polygons in the front and back of the model. So you may end up selecting polygons you didn't mean to. Turn the model to double-check your selection.

Notice the left window of the PEP Texture Applicator. This is a flat mapping of the texture and the polygons it covers. Switch to this window to choose a part of your selected polygons and move, resize, or rotate the texture. See [Remapping a Texture in the PEP Texture Applicator](#) for detailed steps.

Jump to: [Fine-tuning a Texture](#)

Remapping a Texture in the PEP Texture Applicator

To remap a texture:

1. Make sure you have an area of polygons selected (see [Selecting Polygons in the PEP Texture Applicator](#)).
2. In the left window (the 2D view of the texture) select the button pertaining to the action you want to perform on the selected polygons.

The arrow lets you select points of the polygons you have already selected in the left window (it's a sub-selection).

The next two buttons work in conjunction with the square orange box, which is the center of rotation and scaling.

3. Move the center point by selecting its button or by pressing Ctrl-left mouse button and clicking anywhere on the texture map. Then choose the scaling button or the rotation button, and drag the mouse in the left window. You'll see the texture move with the selected points.

You can go to the preview window (the right window with the 3D view) at any time to see what the changes will look like.

4. When you like the way the texture looks, press the *Apply* button.

Jump to: [Fine-tuning a Texture](#)

PEP Texture Applicator

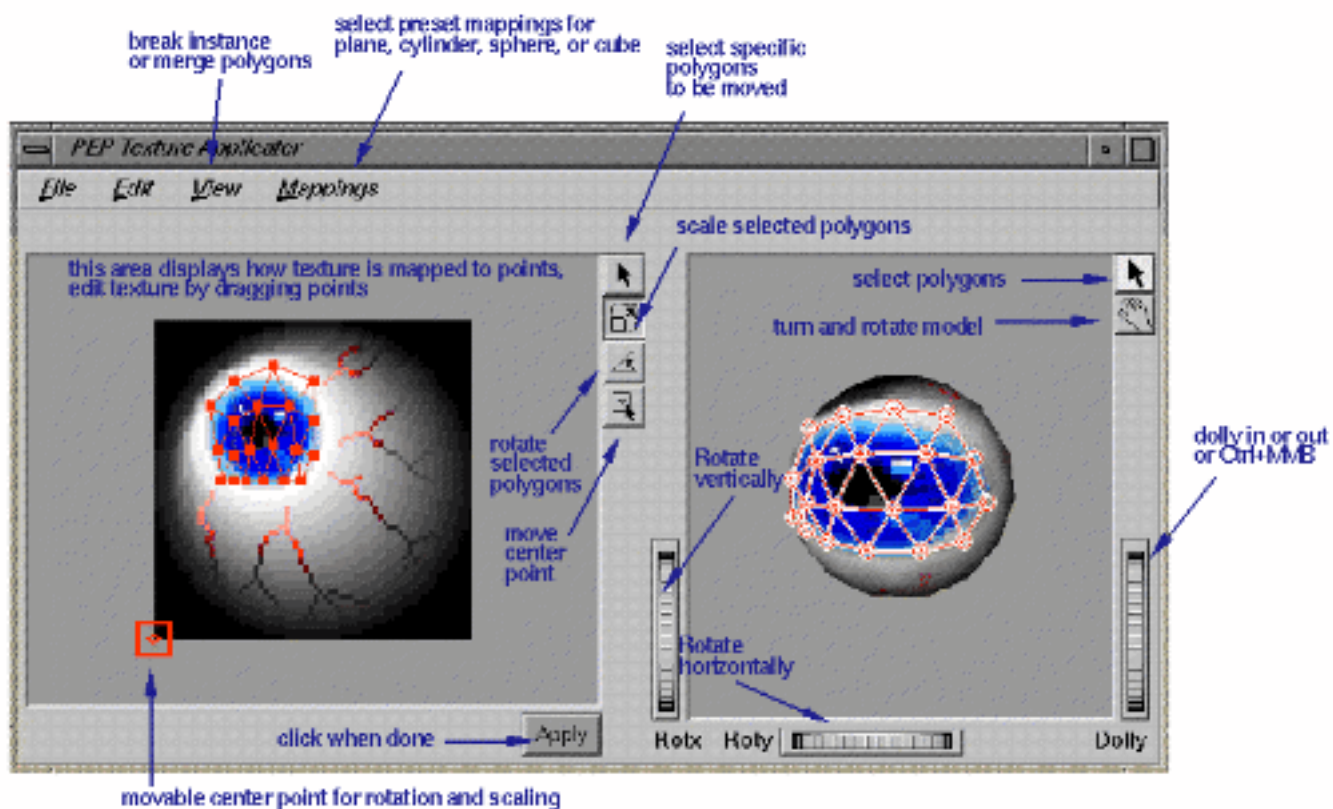


Find it: Click its button on the *Looks* palette:

Use to:

- Fine-tune textures to get rid of seams on PEP objects.
- Break off instanced polygons to remap texture.
- Specify exactly how the texture gets mapped onto a face of an object.

How It Works



[Click image for full view.](#)

For step-by-step information on using the PEP Texture Applicator, see [Fine-tuning a Texture](#).

The Menu Bar:

- **File**
 - *Interactive Change*--turn off if rendering in the preview window is slow. Default is on.
 - *Close*--remember to click *Apply* to save your changes before closing the PEP Texture

Applicator.

- **Edit**

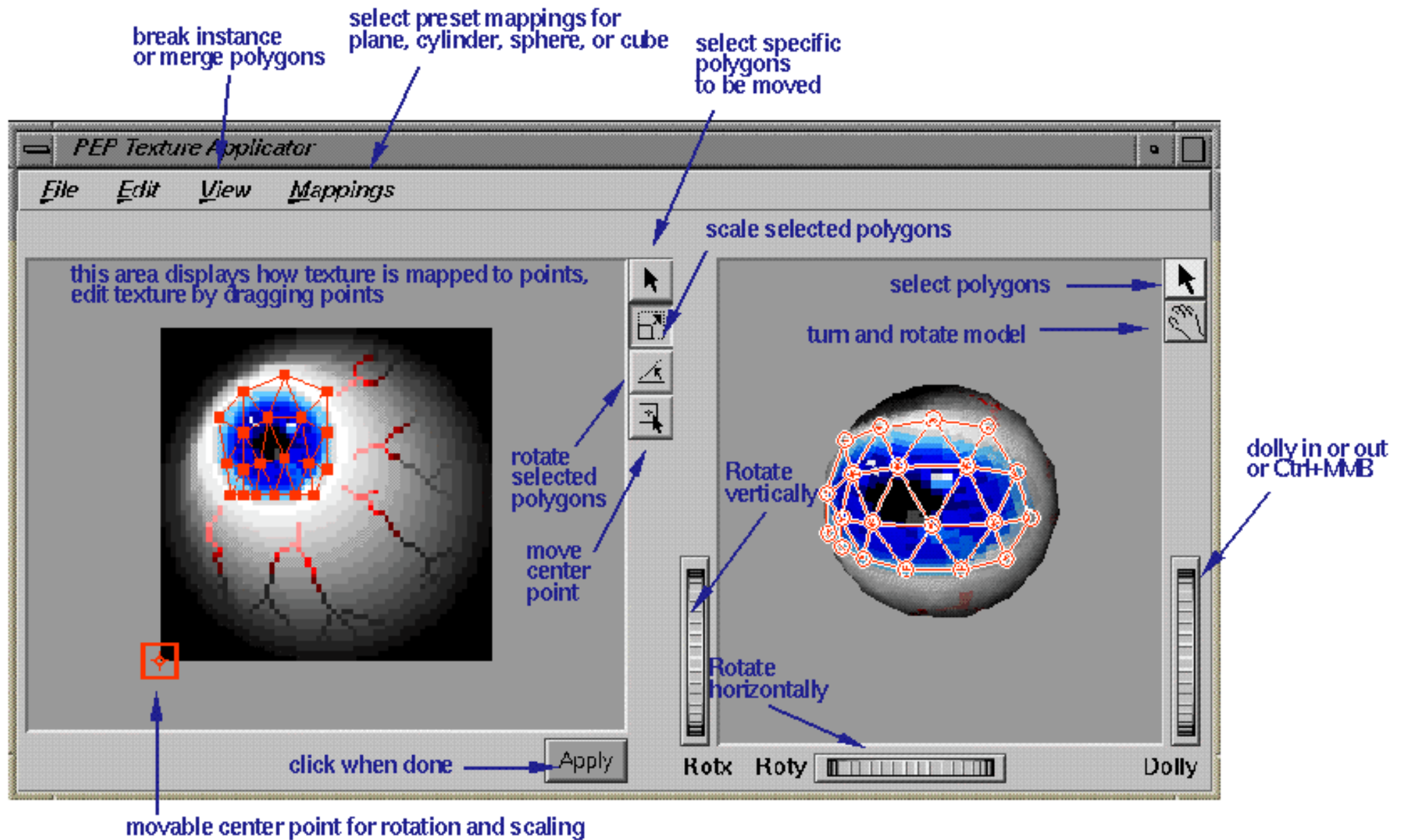
- *Undo* and *redo* of local operations.
- *Break Instancing*--breaks sharing of texture coordinates between polygons. This lets you change the texture mapping on selected polygons only.
- *Merge Selection*--makes the selected polygons share texture coordinates where they coincide.

- **View**

- *View Selection*--zooms in to mapping of selected polygons
- *View All*--zooms in or out to mapping of all polygons in object
- *Auto View Selection*--interactively zooms as you select polygons


- **Mappings**--use to get best texture mappings for object shape. By default, the [Texture Editor](#) maps the texture to the most prevalent surface of a model and smears the texture across the rest of the model. Using *Mappings*, you can change this mapping to reduce or eliminate smearing. For example, to get the best texture mapping for a chess piece, choose *Cylinder: Best*. You can also select mappings specific to X, Y, and Z combinations.

Mapping applies to the current polygon selection--if nothing is selected, it applies to the entire object. This enables you to map different functions to different parts of an object.

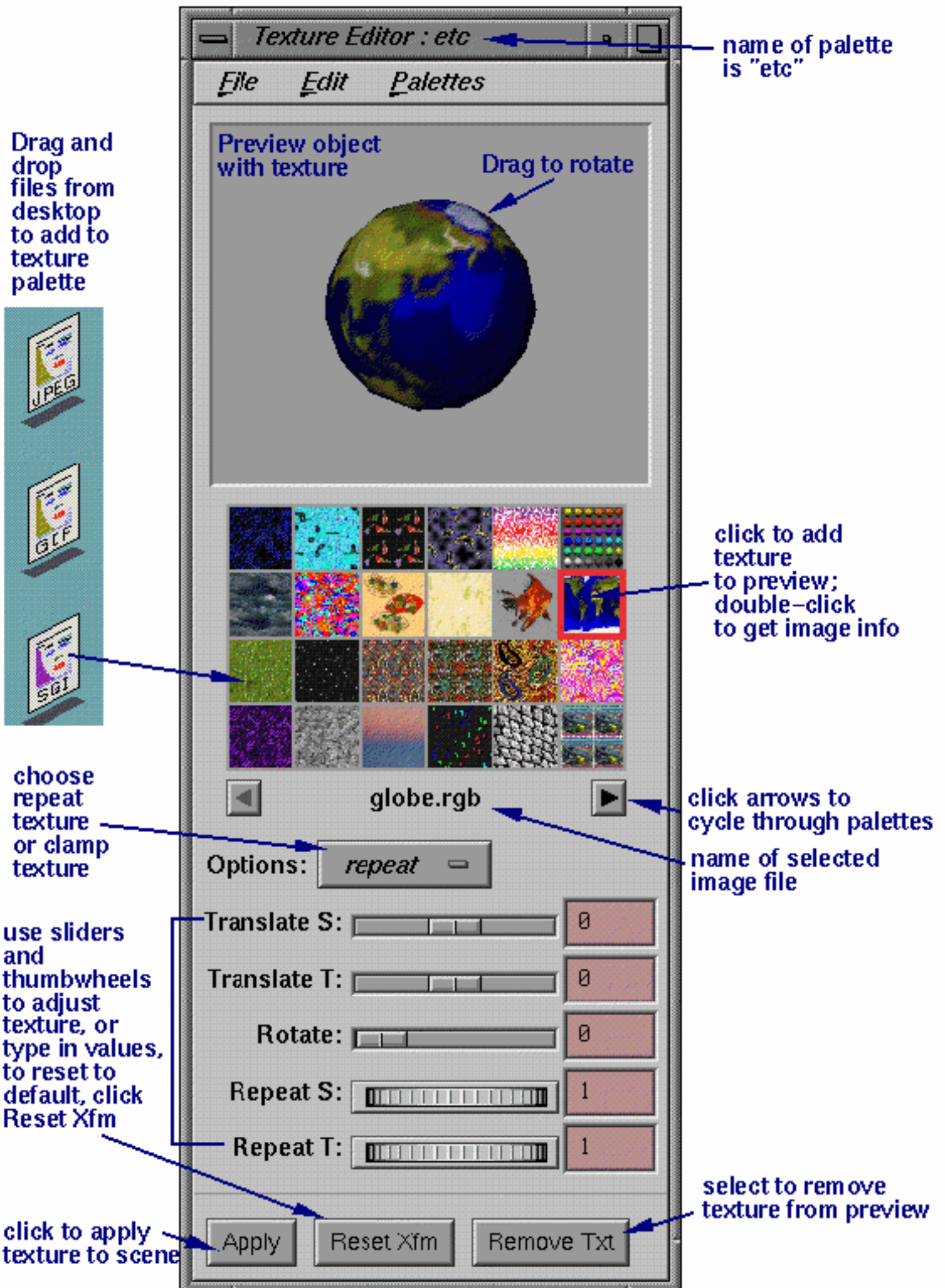


Creating New Texture Palettes

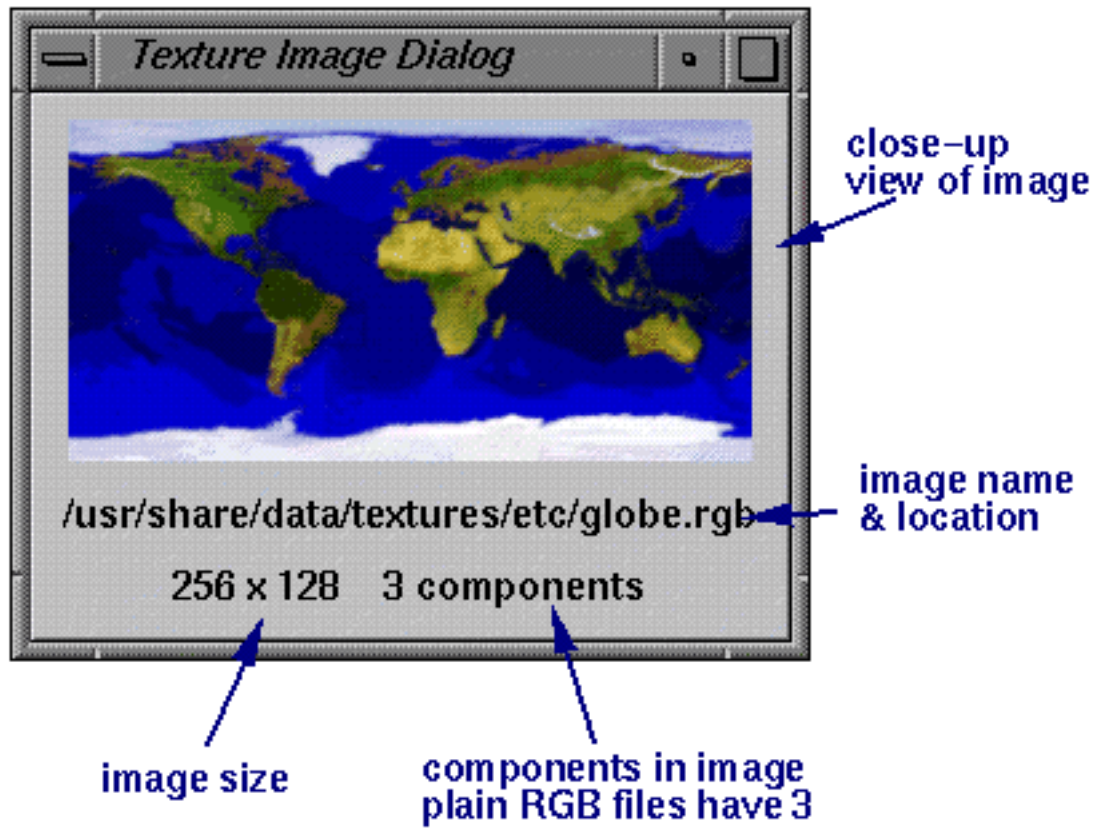
Find it: *Looks palette > Texture Editor*

1. Open the [Texture Editor](#). 
2. Choose *File > New Palette*. A dialog box appears.
3. Enter a name for the palette you're creating and click *OK*. A new, empty palette appears.
4. Choose *File > Import Image*, or drag-and drop an image from your desktop. You can import GIF, JPEG, PNG and RGB images to use as textures.

Jump to: [Applying Textures](#)







Creating Links

on this page: [creating](#) | [deleting](#) | [linking to a viewpoint](#) | [editing](#) | [advanced](#)

Find it: Click the Link Editor button on the *Action* palette:



Use the Link Editor to add hyperlinks to the objects in your scene. What happens when you click a linked object depends on the type of target file specified. If the target URL is a VRML file, clicking the linked object replaces the current scene in the browser with the new scene. If the target URL is a sound file, clicking the linked object plays the sound. If the target URL also contains a viewpoint, clicking the linked object moves you to the specified viewpoint within that scene.

Creating a Link

To create a link to another document (a VRML scene, an HTML document, or a sound file, for example):

1. In the main window, select the object that will contain the link.
2. Open the Link Editor.
3. In the URL text box, type the URL. For example:

```
http://beanbag.esd/usr/people/josie/amusementPark.wrl
```

To browse files, click the *Browse* button. To view URLs specified previously, click the arrow button.

4. The *Description* field allows you to specify a user-friendly description of the link. The browser may display this description in place of the target URL when the user moves the mouse over the linked object. For example:

```
Click to travel to the Amusement Park.
```

5. The *Parameters* field is used by certain browsers that interpret additional information for hyperlinks. For example, some HTML browsers allow you to specify a target frame within the document. When the HTML file is loaded, it is initially placed at this target frame.

To name a specific frame in an HTML document, use the syntax `target=` and then specify the name of the frame. For example:

```
target=ContentsFrame
```

6. Click the *Create Link* button.

Deleting a Link

To delete a link:

1. In the main window, select the object that contains the link.
2. In the Link Editor, click the *Remove Link* button.

Linking to a Viewpoint

You can create links to named viewpoints, both in the current scene and in other scenes. This feature allows you to click objects in the scene and automatically travel to a new viewpoint.

To create a link to a viewpoint in the current scene:

1. Use the [Viewpoint Editor](#) to create one or more named viewpoints.
2. In the main window, select the object that will contain the link.
3. Open the Link Editor. Place a check mark in the *Current Document* box if it is not already checked.
4. Specify the target viewpoint in the *#Target in URL* box. Use the *Browse* button to browse viewpoints defined in the current scene.

For example: #EntryView

Cosmo Worlds will insert the # sign if you don't include one.

5. Click the *Create Link* button.

To link to a named viewpoint in a different scene, you need to know the name of the viewpoint you want to link to. (There is no browser facility for searching viewpoints in other scenes.) Determine the viewpoint name by looking at the VRML file with the Outline Editor.

To create a link to a viewpoint in a different scene:

1. In the main window, select the object that will contain the link.
2. Open the Link Editor. Be sure the *Current Document* box is not checked.
3. Type the URL for the target document into the *URL* field. For example:

`http://beanbag.esd/usr/people/josie/amusementPark.wrl`

4. Type the # sign and the viewpoint name in the *#Target in URL* field:

#EntryView

Editing an Existing Link

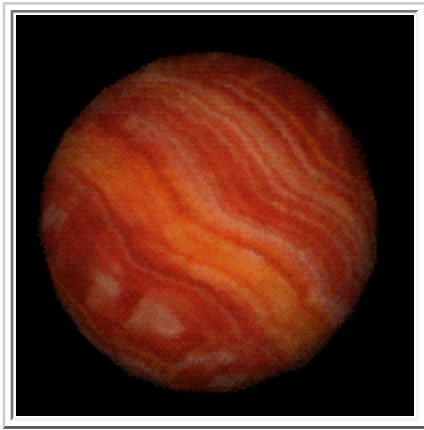
To edit an existing link:

1. In the main window, select the linked object.
2. Open the Link Editor and edit any of the text boxes.
3. Click the *Change Link* button.

Advanced: The Link Editor adds an Anchor node to the scene file. The Anchor node is placed above the current selection (the object containing the link). The object containing the link is a child of the Anchor node.

Jump to: [Defining Viewpoints](#)

Part 1: Making a Planet



The first part of this tutorial introduces you to the Cosmo Worlds user interface. You learn how to create a simple shape, change your view of an object in the scene, and alter an object's appearance. Part 1 includes the following sections:

- [Opening and Saving](#) ↗
 - [Undoing Mistakes](#) ↗
 - [Creating a Sphere](#) ↗
 - [Applying a Texture](#) ↗
 - [Viewing the Planet](#) ↗
 - [Finishing Up](#) ↗
-

Opening and Saving

1. If Cosmo Worlds is not open, open it now by double-clicking its icon:



If you cannot find its icon, choose *Find > An Icon* from the toolchest menu and type *cosmoworlds*. Press *Enter* to open the application, or drag its icon to your desktop and double-click the icon.

2. Save the file as *planet.wrl*: Choose *File > Save As* and type in *planet.wrl*. Now you can use *Ctrl-s* the next time you save the file.

Undoing Mistakes

If you make a mistake, undo it by pressing *Ctrl-z* or choose *Edit > Undo*. To redo, press *Shift-Ctrl-z*

or choose *Edit > Redo*. You can undo and redo as many times as you like, up to the point where you last saved.

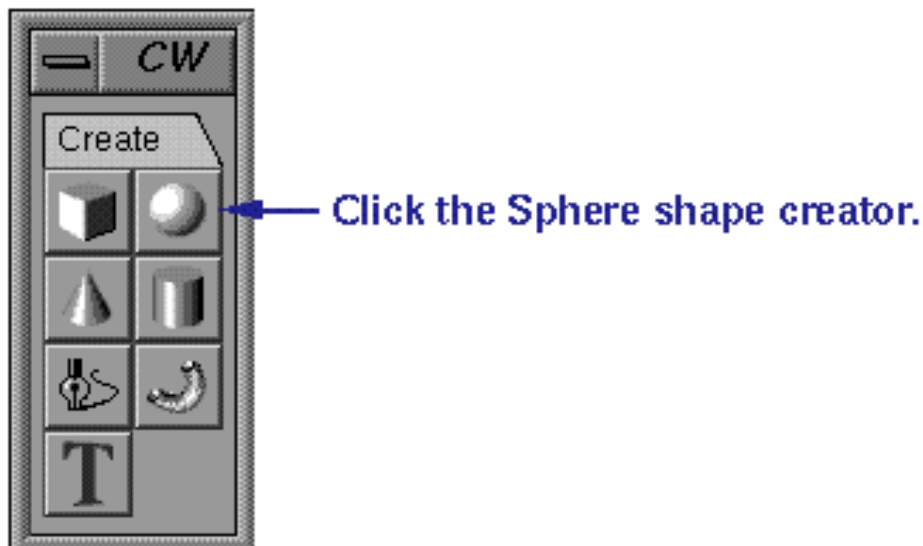
Creating a Sphere

If you haven't done so already, take a moment to examine the Cosmo Worlds interface. It consists of a work and view area (the black area), a menu bar, a customizable toolbar, viewer buttons along the right side of the window, viewer thumbwheels along the bottom and side of the window, and several palettes. If the palettes are not open, you can access them from the *Palettes* pull-down on the menu bar.

Jump to: [A Quick Look at the User Interface](#) to learn more about this topic.

For now, create a sphere.

1. On the *Create* palette, click the Sphere icon:



2. Place the mouse cursor over the work area in Cosmo Worlds main window. Drag the cursor in an outward direction and release. You have just resized and placed a sphere.

Jump to: [Placing a Basic Shape](#) for more information on this task.

Applying a Texture

1. Make sure you are in pick mode for selecting and editing objects. The arrow icon among the viewer buttons (on the right side of the main window) should be highlighted:



If it is not highlighted, click the arrow icon now.

2. Select the sphere by clicking it. An Object Manipulator appears around the sphere. The manipulator indicates a selection. It is also used to resize, rotate, and move the object which it surrounds.

3. Open the Texture Editor by clicking its button on the *Looks* palette:



4. Choose a texture. When you click a texture, you see a preview of the sphere with the texture. When you find a suitable planet texture, press *Apply* to place the texture on your sphere.

Jump to:

- [Texture Editor](#) for more information on this tool.
- [Selecting and Grouping Objects](#) or [Moving, Resizing, and Rotating](#) for more information on these tasks.

Viewing the Planet

Before you continue, take a moment to examine the planet you've just made.

1. Make sure you are in view mode. The hand icon among the viewer buttons (on the right side of the main window) should be highlighted:



You can also temporarily switch to view mode by holding down the *Alt* key.

2. Drag over the scene to rotate the camera (your view) around the planet. Drag quickly and release to make the planet spin; click to stop the spin.

Jump to: [Viewing Objects and Scenes](#) for more information on this task.

Finishing Up

OK, you're done with the first part. Save the file by clicking *Ctrl-s* and continue to Part 2 without exiting. If you'd like to go on to Part 2 later, then exit from the program by choosing *File > Exit* or use *Ctrl-q*. Now, on to something new.

[Part 2: Creating a Rocket](#) 

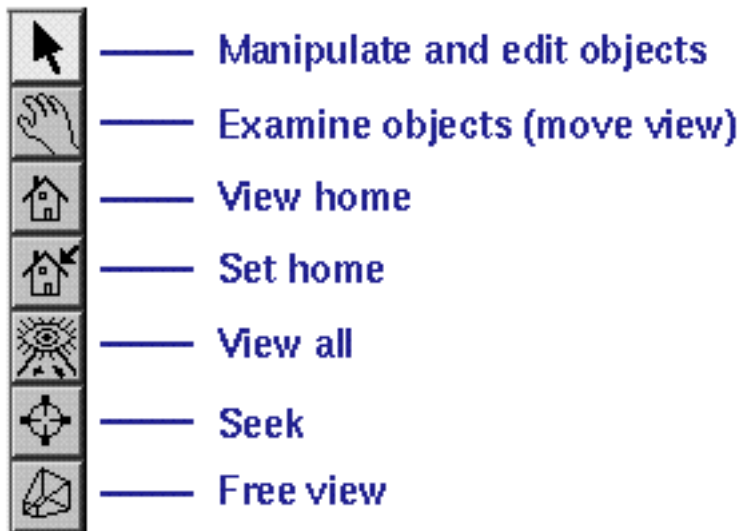
A Quick Look at the User Interface

on this page: [viewers](#) | [menus](#) | [palettes](#) | [toolbar](#)

Viewers

- [Examiner Viewer](#): use for rotating, panning, zooming to examine objects.
- [Walk Viewer](#): use for navigating through a scene.
- [Plane Viewer](#): 2D viewer used in some editors and fixed views (top, left, right, etc.)

Viewer Buttons



Also see [Tools to Use: Viewing](#).

Grouping Buttons



When you select an object that is grouped, the Parent or Child buttons become visible.

- **Parent:** Press Parent to select the top-most level of the grouping.
- **Child:** Click on the part of the object you'd like to select, then press Child until that part becomes selected (manipulator handles appear on the selection).

See [Selecting and Grouping Objects](#) for more information.

Menus

File Menu

- **New:** clears the window to create a new scene.
- **[Open:](#)** opens and reads a scene saved in Cosmo Worlds and related formats
- **[Import:](#)** imports objects from a specified file.
- **[Import as Inline:](#)** creates an inline to include distributed objects referenced by URL.
- **[Interactive Import Placement:](#)** check this option ON to interactively place and resize an imported object; deselect (turn checkmark OFF) to retain object's original placement.
- **Open Gallery View:** open Cosmo Worlds gallery, a collection of sample VRML files for use in your own scenes.
- **Save:** saves the scene in Cosmo Worlds format (VRML 2.0 plus special authoring nodes).
- **Save As:** saves the scene under a new filename.
- **Revert:** opens the last saved version of the current file.
- **Preview:** launches Cosmo Player with the current scene loaded for viewing and interaction.
- **[Package:](#)** launches Cosmo Package, a program that locates all files included by your document, scene, or world, and organizes the files for publication on a server. This process strips authoring source nodes from your scene to produce a strictly compliant VRML 2.0 file.
- **Save Layout:** retains current layout for next session
- **Exit:** exits Cosmo Worlds.

Edit Menu

- **Undo:** undoes each previous operation in a linear succession. You may "rewind" back any number of steps until the last, previous file operation (i.e. open, save, etc.); until the first action *after* a previous undo sequence; or until you have exhausted the memory of the Cosmo Worlds authoring system.
- **Redo:** redoes what was just undone, if applicable.
- **Edit Object:** launches the editor appropriate for modifying the currently selected object. For example, selecting an extrusion object, then choosing Edit Object will open the Extrusion Editor.
- **Cut:** removes the selected object from the scene, and retains it in the clipboard for further use.
- **Copy:** copies the selected item into the clipboard for further use.
- **Clone:** defines the selected object as a master for producing multiple instances. All instances share this master description identically, tracking all changes to its shape and appearance. Use Paste Clone to place a new instance of this cloned object in the scene.
- **Paste Copy:** pastes a new copy of objects from the clipboard into the scene.
- **Paste Clone:** pastes a new instance of a cloned object into the scene (see **Clone** above).
- **[Interactive Paste Placement:](#)** check this option ON to interactively place and resize a pasted object; deselect (check OFF) to retain the object's original placement.
- **Group:** collects selected objects under a new parent group node, [making a hierarchy](#).

- **Ungroup:** disbands a selected parent object. The selected node is removed and its constituent children become top-level objects in the scene.
- **Add to Group:** inserts the selected object into the selected group.
- **Detach from Group:** removes a selected object from its parent group and add it as a top-level object to the scene without ungrouping the rest of that group.
- **Create Parent Group:** creates a new parent group over the selected object, retaining its place in the scene hierarchy. This is a good way to insert a dedicated transform node, for example, in applying a jointed animation.

Select Menu

- **Select All:** selects all top-level objects in the scene.
- **Deselect All:** deselects all objects in the scene.
-
- **Select Highest:** selects the highest level parent group that contains the picked objects will be selected.
- **Select Lowest:** selects the lowest level object picked.
-
- **Select Parent:** selects the immediate parent of the current selection.
- **Select Child:** selects the next lower child object within the selected [group in the hierarchy](#).

View Menu

- **View All:** moves the camera so all objects are in view.
- **View Selection:** moves the camera for a close view of the selected object. This is very helpful when a selected object is not in direct view.
- **View Home:** positions the camera to show a view from the home position.
- **Set Home:** sets the current view as the home position.
- **Hide Selected Objects:** hides selected objects from view.
- **Hide Unselected Objects:** hides all unselected objects from view.
- **Show Selected Objects:** shows all selected objects that were hidden by choosing Hide Selected Objects
- **Show All Objects:** displays any objects that were hidden using Hide Selected/Unhide Selected.
- **Show/Hide Iconic Objects:** toggles between hiding or showing all [iconic objects](#) in the scene; these include objects like sounds and local viewpoint cameras.
-
- **Display Polygon Count:** lets you keep track of the polygon count for a single object or for all objects in the scene.
- **Nice Transparency:** renders objects in a transparent, high quality manner.
- **Display Fog:** toggles between showing and hiding a fog node added to the scene.
- **Display Background:** toggles between showing and hiding a background node added to the scene.

-
- **Set Working View:** choose from a perspective projection ("free" view) and construction views, which limit the camera to a single plane and an orthographic projection.
 - **Examiner Viewer:** activates the Examiner Viewer mode, in which the scene is rotated as if it were a ball held in your hand.
 - **Walk Viewer:** activates the Walk Viewer mode, in which the scene is traversed as if walking.

Layout Menu

- **Nudge:** use arrow keys or the pull-right menu to move a selected object in increments.
- **Activate Snap Target:** places the yellow Snap Target in the scene.
- **Activate Snap Source:** places the green Snap Source arrow in the scene.
- **Activate Scale Snapping:** helps you scale desired features on a selected object (for example, a cylinder's height) to the same measurement as that of an object on which the Snap Target is placed.
- **Center Snap Target in Selection:** centers snap target in the middle of the nearest plane of the selected object.
- **Move Selection to Snap Target:** moves a face of the selected object to the Snap Target.
- **Move Selection Center to Snap Target:** moves the selected object's center to the Snap Target.

-
- **Activate Grid Layout Tool:** places a rectangular grid in the scene. Drag the snap target onto this grid to arrange objects.
 - **Activate Angle Layout Tool:** places a concentric grid in the scene. Drag the snap target onto this grid to arrange objects.
 - **Precision Placement:** opens a panel where you can type in numeric values to position, orient, and scale the selected object.

-
- **Grid Layout Tool Options:** set options for rectangular grid.
 - **Angle Layout Tool Options:** set options for circular grid.
 - **Snapping Options:** set options for controlling snapping behavior when using the Snap Target.

Palettes Menu (explained in next section)

Palettes

Find it: Choose *Palettes > Creators, Editors, etc.*

The palettes include:

- [Create Tools](#)
- [Editors](#)
- [Media Tools](#)
- [Action Tools](#)
- [Layout Tools](#)
- [Viewing Tools](#)
- [PEP Tools](#)
- [Split Tools](#)
- [Looks Tools](#)

To use the palettes: Click on a button to perform an action. A short description of each button appears below the menu bar when you pass your cursor over the button.

To customize the palettes:

- Collapse and expand palettes by clicking their heading names; these look like tabs.
- Drag the palette heading tabs to:
 - move the palette up or down within the palette window
 - break out the palette into its own window
 - drop the palette into another palette window
- When you drag a heading tab, a blue marker appears. This indicates where the palette that you are moving will appear:



Toolbar

The toolbar is a collection of palette buttons and spacers that you can add and subtract to create your own custom toolbar.

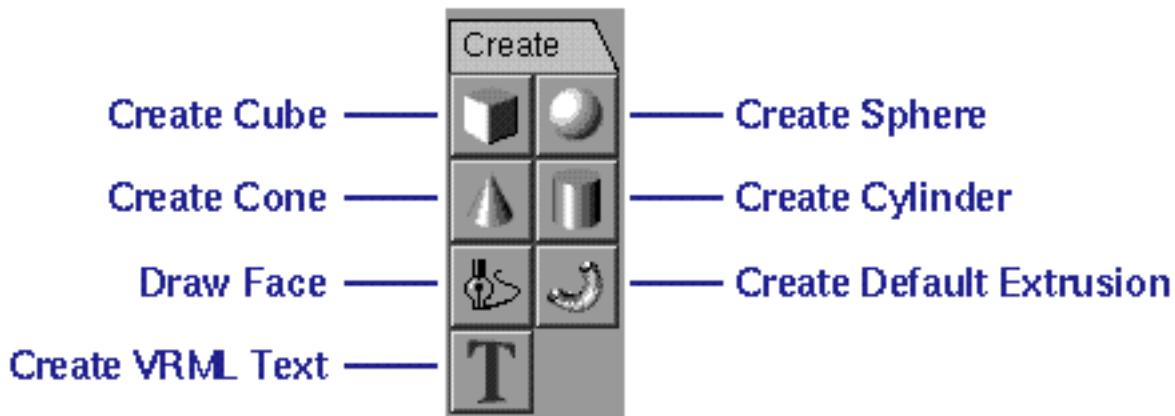
- Drag buttons off the toolbar to remove them. Drag buttons from the tool palettes to add them

to the toolbar. A blue marker indicates where the button will appear:



- Create and remove spaces between buttons by dragging palettes a half-button width to the left or right on the toolbar.

Create Palette



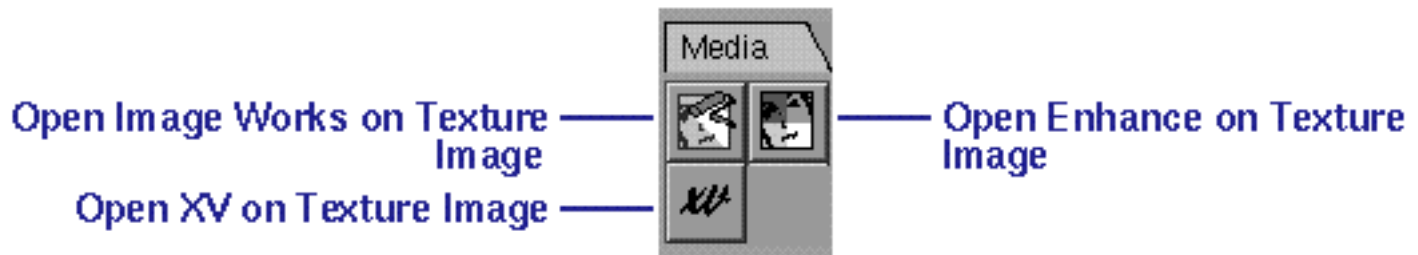
See [Creating Basic Shapes and Text](#).

Editors Palette



Click the text or button for more information on an editor.

Media Palette



The media palette lets you quickly access an application and make changes to textures in your scene. For example, to edit a wood texture you applied to a table, select the table and press the Open Image Works on Texture Image button. Image Works opens with the wood texture ready for editing.

Image Works, XV, and Enhance are default Media plug-ins. To configure other applications as Media plug-ins, see [Adding a Helper Application](#).

Adding a Helper Application

Both Cosmo Worlds and Cosmo Create use a database that contains information on all helper applications they use. This file is installed as */usr/lib/CosmoCreate/plugins/PiHelperAppDB.txt*.

You need to add an entry to this file for each helper application (also referred to as a "plug-in application") that you want to access through Cosmo Worlds or Cosmo Create. The entries in the database file look like this:

```
Begin
  ID: com.sgi.cosmo.photoshop
  UINAME: Photoshop
  REPTYPE: PiUnixApp
  FLAVOR: Editor
  EXECUTABLE: photoshop
  ARGS: -open %f
  PIXMAP: photoshop.xpm
  NATIVE: PhotoshopDocument
  FILETYPES: PhotoshopDocument GIFImageFile TIFFImage SGIImage
End
```

Go to [Helper Application Database File](#) for detailed information on the format of the database file.

For each application, you also need to add a pixmap. The database file contains an entry specifying the relative location of the pixmap for each helper application.

Jump to: [Helper Application Database File](#)

Helper Application Database File

This document describes the purpose and format of the *PiHelperAppDB.txt* file.

Purpose

The *PiHelperAppDB.txt* file is a database containing helper application descriptions. Helper applications are programs that can be launched through mechanisms in the Cosmo Create and Cosmo Worlds plug-in architecture.

Format

The format of the file is line-oriented and very simple. The file should contain zero or more application description blocks, each beginning with "Begin" and ending with "End." Within the block, each line describes a single attribute of the application description. The line consists of the field's ID, followed by a colon and then the value of the attribute.

```
Begin
  ID: com.sgi.cosmo.photoshop
  UINAME: Photoshop
  REPTYPE: PiUnixApp
  FLAVOR: Editor
  EXECUTABLE: photoshop
  ARGS: -open %f
  PIXMAP: photoshop.xpm
  NATIVE: PhotoshopDocument
  FILETYPES: PhotoshopDocument GIFImageFile TIFFImage SGIImage
End
```

Field Meanings

- **ID:** the unique string ID of the helper application.
- **UINAME:** the name that should appear in the user interface to represent this application.
- **REPTYPE:** the name of the class that should represent this application at runtime. Currently, only "PiUnixApp" is supported here. A PiUnixApp is started with some forks(), and FAM (the File Alteration Monitor) is used to monitor changes to the file being edited.
- **FLAVOR:** specify "Editor."
- **EXECUTABLE:** this is the executable to run. It may be an absolute path. If it is relative, the user's path is searched.
- **ARGS:** the arguments to pass the program when running it.
- **PIXMAP:** the filename of an XPM file containing the pixmap for this helper application.
- **NATIVE:** this is the FTR name of the application's "preferred" file type -- the type that should

be created if you need to make a new file of this type. This field should have only one value. This value should be included in the FILETYPES field.

- **FILETYPES:** these are the FTR names of the file types the application will handle. This list should include the name in NATIVE. **Note:** you do not have to put *all* of the file types that your application handles, just the ones you are willing to have the database offer it for.

PiUnixApp launching

PiUnixApp uses the EXECUTABLE and ARGS fields of its application description when it launches the application.

Since the application is used by a PiEditSession to edit a file on the disk, the arguments can be parameterized by information about that file (similar to a mailcap file). The basic input is the file to be operated on. Different parameters are available to provide information about different parts of it.

File-Centric Parameters

- %D -- the directory of the file to be edited
- %f -- the filename of the file to edit (without directory)
- %F-- absolute path to the file to edit
- %copy -- filenames of the input copy of the file (without directory)
- %COPY -- absolute paths to the input copy file

Some programs require both an input and an output file. We provide %copy and %COPY for these programs. If the PiUnixApp sees one of these parameters, it copies the input file to a temporary file in the same directory and sets %copy and %COPY appropriately.

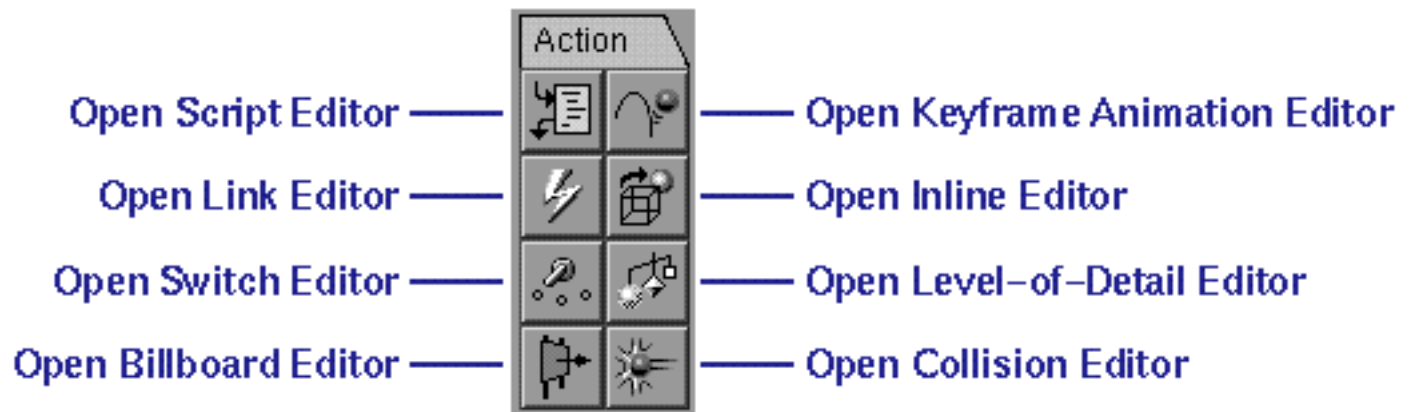
Window Mapping Parameter

- %wr -- for window role property. This will be replaced by something like

```
-xrm *windowRole: PiUnixAppXXXXXXX
```

Put %wr in your args only if you support *-xrm* args. You probably support them if you pass your arguments to XtInitialize.

Action Palette



Click the text or button for information on that editor.

Creating Billboards

Find it: Click the Billboard Editor button on the *Action* palette:



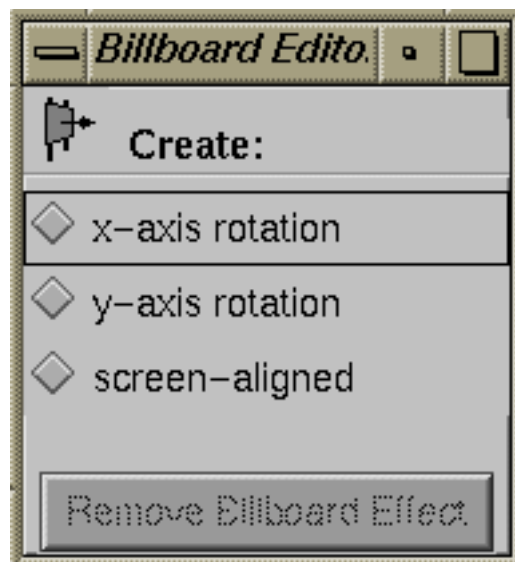
The Billboard Editor lets you create objects that rotate around a specified axis and always face the viewer. Billboards are useful for optimizing the scene, since you can use 2D objects to simulate 3D objects and save large numbers of polygons. For example, you can import an image of a tree and specify that it rotates around its y axis. Since the tree image is always facing the user, he or she won't be able to see that it's simply a flat image.

Most commonly, billboards rotate around the y axis. For example, objects such as lampposts and trees rotate around the y axis. Objects that rotate around the x axis tip to face you as you fly over them. A ferris wheel cage is an example of an object that rotates around the x axis.

Billboards are also useful for creating *screen-aligned* objects such as text labels and icons.

To create a billboard, follow these steps:

1. Click the Billboard Editor button on the *Action* palette.
2. Select the object in the scene that you want to make into a billboard.
3. Click the button for the type of rotation desired: x axis, y axis, or *screen-aligned*.



To edit an existing billboard, open the Billboard Editor and click the button for a new rotation. To convert the object back to its original, nonrotating form, press *Remove Billboard Effect*.

Jump to:

- [Optimizing the Scene](#)

modapprchPlan Your Modeling Approach

- Model the individual elements of your scene, and the scene itself. As much as possible, store individual elements (models) of your scene separately. (appropriate for CC3d?)
- Convert splines to polygons early-on. Use polygons to begin with whenever possible. (or is it easy in cc3d to convert splines to polygons?)
- For each element in your scene, design three (or more) versions of the model at varying degrees of complexity. Incorporate these into LODs. Use these general guidelines for the three models:
 - **1000 polygon count, or texture mapped.** Suitable for viewing on a RealityEngine class machine (good polygon performance, hardware accelerate texture mapping).
 - **100 polygon count, not texture mapped.** Suitable for viewing on a Indigo class machine (moderate polygon performance, no hardware accelerate texture mapping).
 - **10 or less polygon count, not texture mapped.** Suitable for viewing on a PC or Mac class machine (low polygon performance, no hardware accelerate texture mapping).

rightnodesUse the Right Nodes

- Edit the vrml scene, adding LOD nodes. (how do you plug-in LODs in cc3d?)
 - Specify an empty Group node as the last child if it is OK for the object to dissappear entirely when it is very far away. No name hint is necessary for the invisible child, if it exists.
 - Wrap all but the invisible and lowest level of detail children in a WWWInline node.
 - Use the `vrmlinfo` utility, shipped with the WebSpace distribution, to count the polygons in an object, and do determine the bounding box info needed by the WWWInline node.
- Use WWWAnchor nodes to wrap all hyper-linked objects. Specify a URL in the name field of the WWWAnchor node.

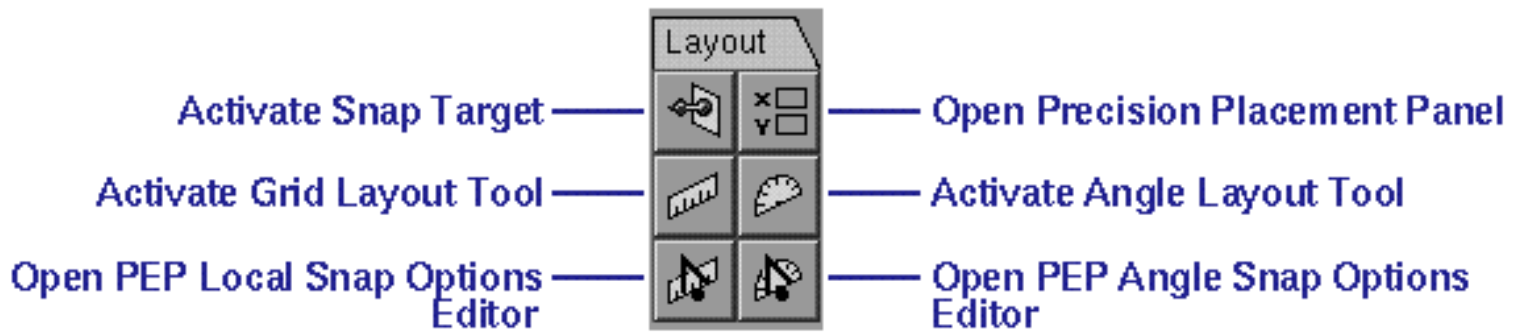
bigpicKeep the Big Picture in Mind

Be conscious of the [overall scene complexity](#).

pubvrmlPublish to VRML

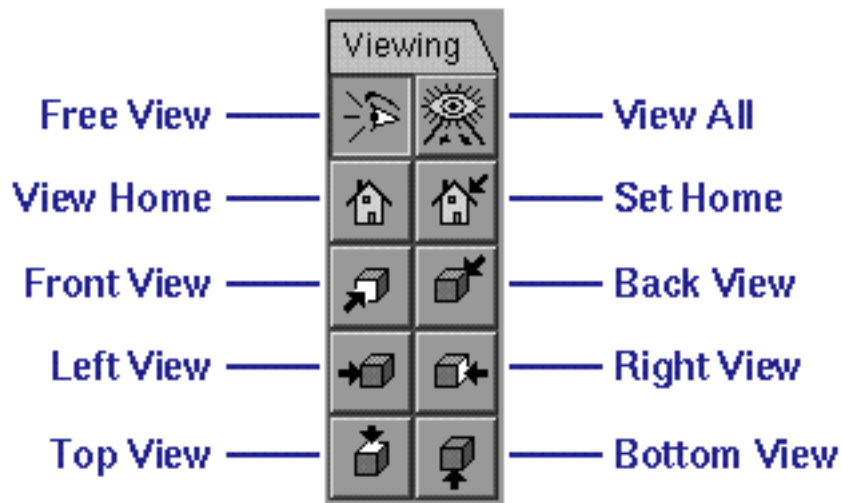
- Use the `ivToVRML` utility to convert from Inventor files to VRML files.

Layout Palette



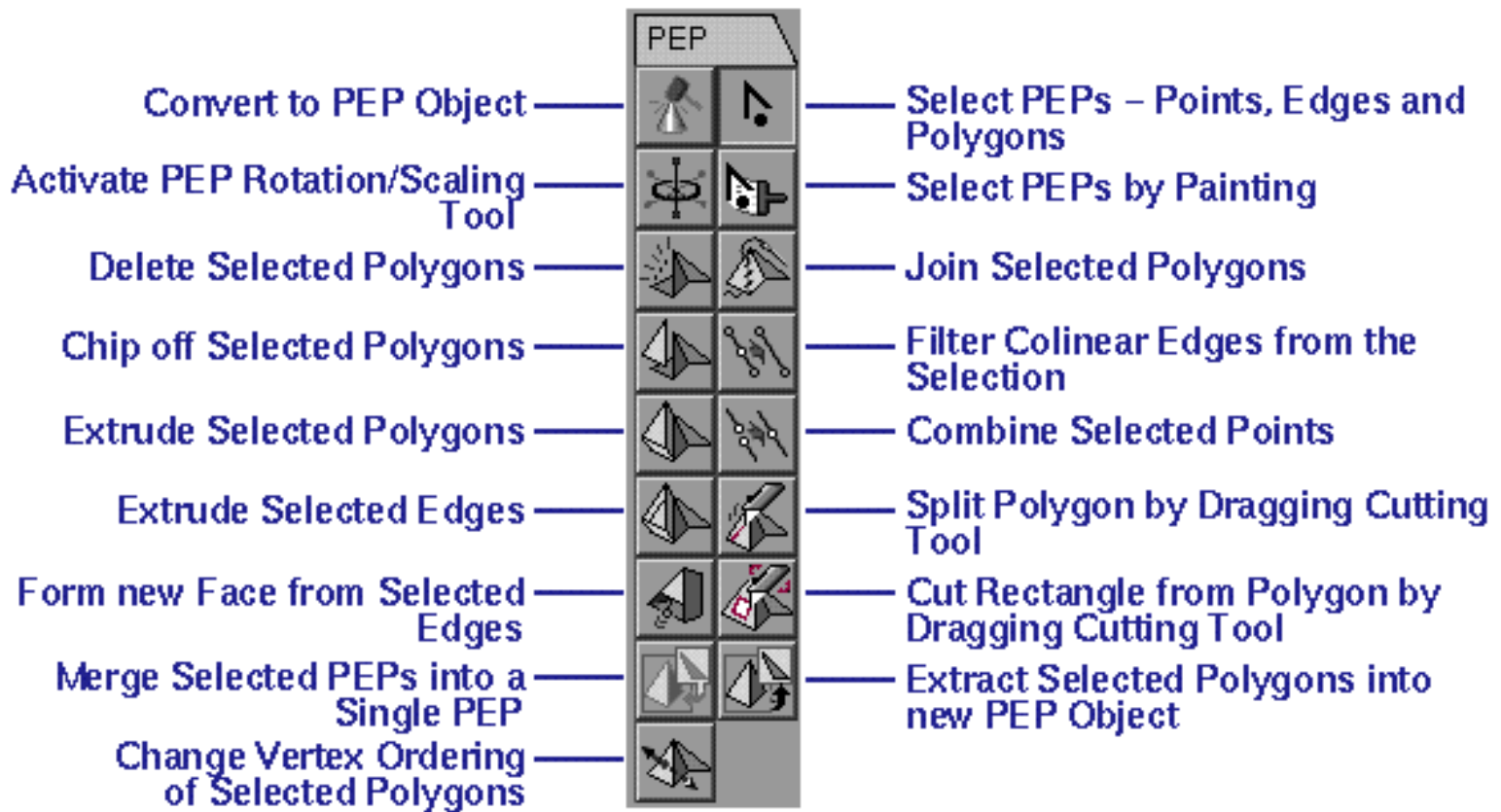
Click the text or button for more information on an item.

Viewing Palette



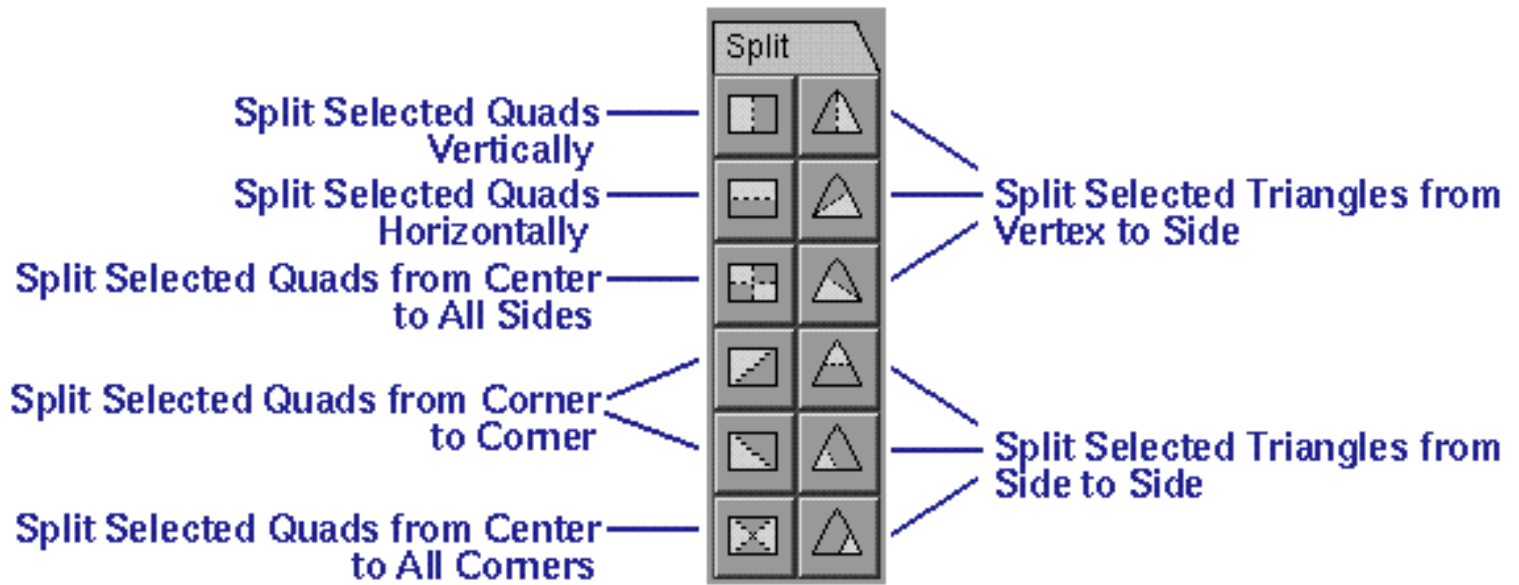
See [Tools to Use: Viewing](#).

PEP Palette



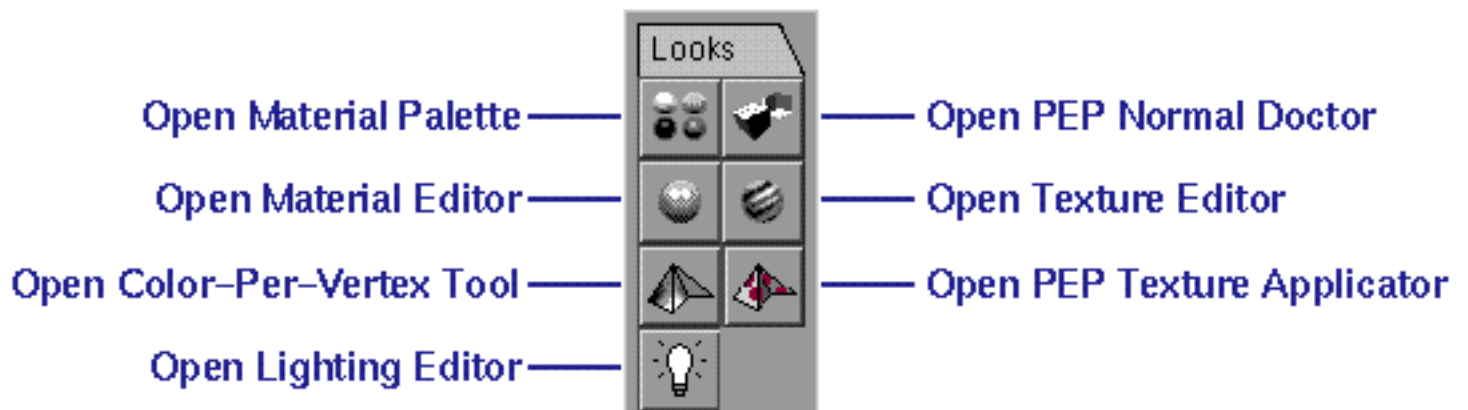
Click the text or button for more information on an item.

Split Palette



See [Splitting and Cutting Polygons](#).

Looks Palette



Click the text or button for more information on an editor.

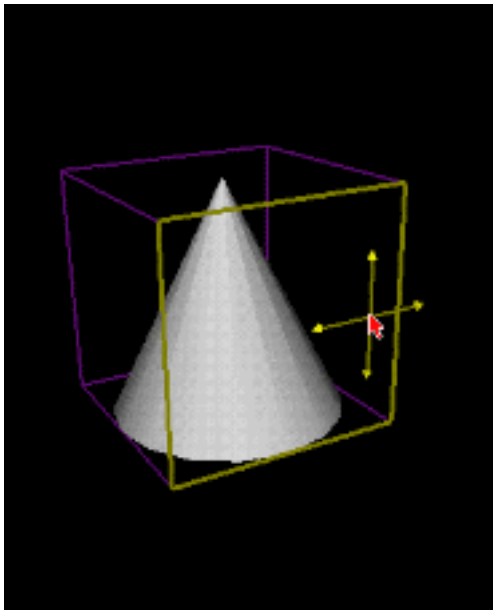
Moving, Resizing, and Rotating Objects

on this page: [move](#) | [resize](#) | [rotate](#) | [move center of rotation](#) | [multiple objects](#)

When you select an object (click it), a box with white cubes and green balls appears around the object. This is the object's *manipulator*. The manipulator gives you area and handles for moving, resizing, and rotating objects. Drag the box to move an object, drag the white cubes to resize an object. or drag the green balls to rotate an object. When you select multiple objects, the last object selected has a manipulator; empty boxes with no handles appear around the other objects in the selection. Moving, resizing, and rotating affects all objects in the selection.

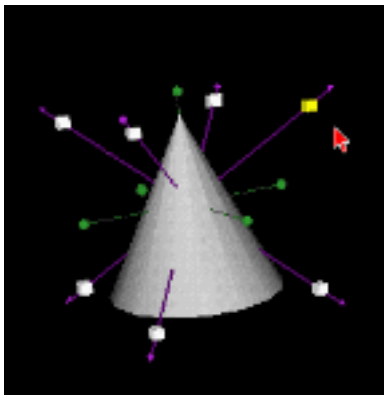
To move:

Click and drag a model's box to move it. *Ctrl* moves model perpendicular to side of box. *Shift* constrains motion in a single direction.



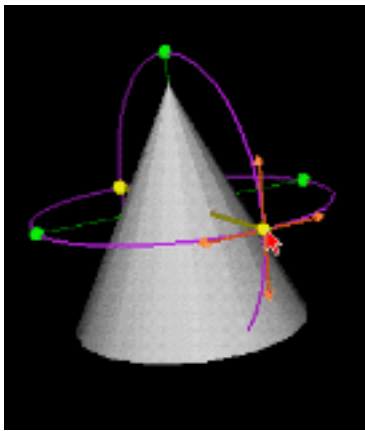
To resize:

Click and drag white cubes. *Shift*-drag for nonuniform scaling. *Ctrl*-drag to uniformly scale about the opposite corner. *Ctrl-Shift*-drag to stretch about the opposite corner.



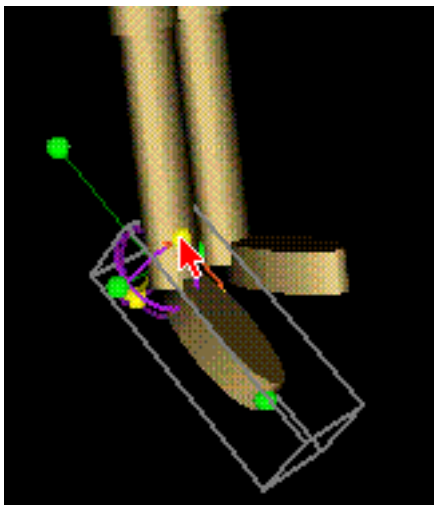
To rotate:

Click and drag green knobs. Movement is constrained by default. *Shift*-drag to unconstrain the rotation.



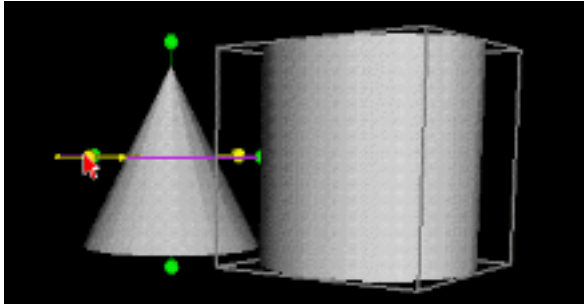
To move object center for rotation:

Ctrl-drag green knobs. This is a useful trick when animating jointed models where, for example, a foot should be rotated at the point to which it attaches to the leg, instead of at the foot's usual center of rotation.



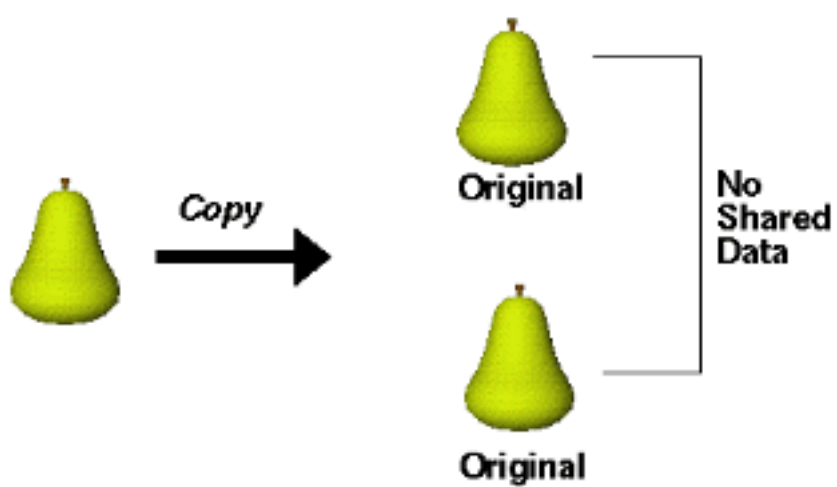
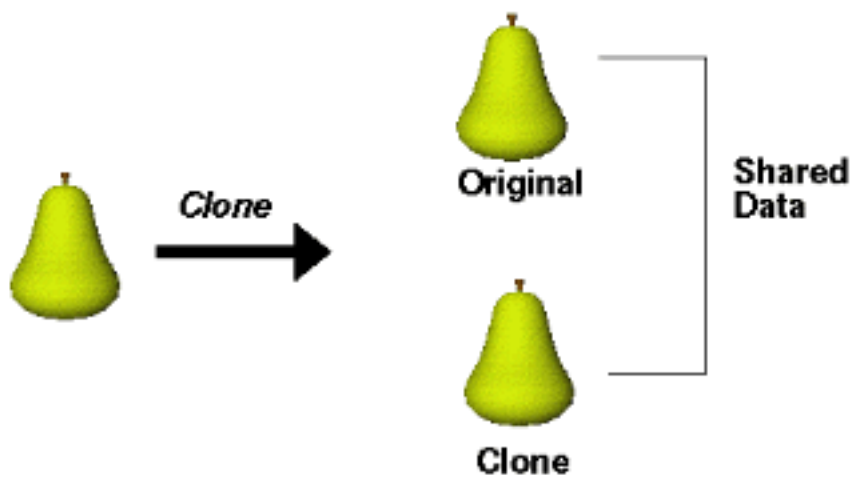
To move, resize or rotate an object relative to another object:

Shift-click to select objects. The last object you select is the master selection--all previously selected objects are transformed relative to the master selection.



Jump to:

- [Manipulating Objects](#)
- [Tools to Use: Manipulating Objects](#)
- [Selecting and Grouping Objects](#)
- [Snapping to Align Objects](#)
- [Layout Tools](#)
- [Precision Placement Panel](#)



Reducing Polygons

Use the [Polygon Reduction Editor](#) to reduce the number of triangles that make up a particular object. An object with a low-polygon count is optimized for viewing on the Web. Also use for making multiple Levels of Detail (LOD) for an object in your scene.

1. Select the object which you'd like to reduce polygons or create multiple Levels of Detail for.

If the object you select exists in the scene as an inline, you can use the Polygon Reducer on it if the object is an *editable inline*. See [Creating Inline Objects](#) for details on creating editable inlines.

If you're creating a LOD, you might want to copy the original object into its own file before going on to the next step.



2. Open the editor by clicking its button from the *Editors* Tool Palette.
3. Choose a quick reduction using *Octahedron* or *Bbox*, or specify a more precise reduction by adjusting the sliders. If you are creating a lowest LOD, use Octahedron or Bbox. This will produce an object with a very low polygon count ideal for viewing at a far distance. For LODs viewed at a closer distance, use the sliders. See [Polygon Reduction Editor](#) for tips on using the sliders for the best results.
4. Click *Accept* to apply the changes to the object in the main view. If you've just created a LOD, see [Creating Multiple Levels of Detail](#) to learn how to implement the LODs in your scene.

Jump to: [Optimizing Your Scene](#)

Keyframe Menu



Find it: Click

The Keyframe Animator *Keyframe* menu has the following options:

Select in range selects all keyframes in all lines within the range. The range is the area defined in the top trough by the range slider. A light gray shading indicates the selected range throughout the time panel.

Select for object in range selects all keyframes for the currently selected objects within the range.

Select none deselects all keyframes.

Cut (*Ctrl* + *X*) removes the selected keyframes from the animation and places them in the cut buffer. No time is removed from the animation.

Copy (*Ctrl* + *C*) copies the selected keyframes and places them in the cut buffer.

The Keyframe Animator has three paste options. All three options use the keyframes currently in the cut buffer.

Paste overlay at current time (*Ctrl* + *V*) pastes the selected keys at the current time. If the keys land at the same time as existing keys, the pasted keys replace the existing keys. If there is no new key to replace an existing key, the existing key is preserved (that is, you'll have a mixture of old and new keys). Use the *Paste replace range* option if you want to wipe out the previous set of keys entirely and replace it with the new ones.

Paste replace range deletes everything currently in the range and pastes in the new range. This option is used to replace a section of animation with a new section. If the Paste selection doesn't fit into the specified range, it is truncated. (Use the *Paste insert at current time* option if you want to insert a selection and make the animation longer.)

Paste insert at current time inserts the Paste selection at the current time, extending the animation the length of the pasted selection.

Delete (Backspace) deletes the selected keyframes.

Snap to frames toggles between snapping to frames (the default) and not snapping, which allows you to place keys at intermediate points on the timeline. When snapping is on, you can drag only to the frames on the timeline, not to intermediate points. You can move the scrub bar only to frames. When

you drag a keyframe, it snaps to the frame. If you type a number such as 3.2 into the *time* text field when snapping is on, it will round to frame 3. When you change the frames per second count and snapping is on, all keyframes are snapped to the new grid. If two keyframes end up on top of each other, only one will remain. (The first selected keyframe will remain. If no keyframes were selected, the one that was earlier in time will remain.) When you change frames per second and snapping is off, the keyframes are left to float at their new positions.

Jump to:

- [View Menu](#)

View Menu (Keyframe Animator)



Find it: Click

The Keyframe Animator *View* menu has the following options:

Show time in frames (the default) displays the time in frames (on the timeline and in the *time* text field).

Show time in seconds displays the time in seconds (on the timeline and in the *time* text field).

For animations under one minute, the format is *ss.ss* (*s* stands for seconds).

For animations between one minute and one hour, the format is *m:ss.ss* (*m* stands for minute).

Limit playback to range allows you to loop through a small piece of a larger animation while you are editing the animation.

The **range slider** is located above the current time trough in a second trough. The slider is a bar that allows you to define a range of keyframes (to select, then cut or paste, for example). To move the beginning or end of the range, drag on the beginning or end of the bar. To keep the range the same length but move it to a different part of the animation, drag on the middle of the bar.

To move the closest end of the bar to the current cursor position, click anywhere in the trough (either to the left or right of the bar, or above it). *Ctrl*+click to move the farthest end of the bar to the current position.

If *Snap to Frame* is on, the range slider snaps to the closest frame. If frames per second changes, the bar resnaps.

Show empty field lines displays all fields--even unused ones--when you expand a property line. Normally, when you expand a property line down to the field level, only fields with animation are shown. Unused fields are not displayed at all unless you enable this option.

Jump to:

- [Names Panel](#)

Names Panel



Find it: Click

This page describes the Keyframe Animator names panel.

The names panel, the column along the left-hand side of the keyframes, lists the *Master*, *member*, *object*, *property*, and *field* components of the animation. The Master and member names (in boldface type) are always listed. To display more information on a member, press the blue down arrow, which displays the objects that compose the member. If you press the down arrow on an object, the properties for the object are displayed (and the object line disappears). If you press the down arrow on a property, the fields for the property are displayed (and the property line disappears). Here's an example of the names for one piece of a robot. Depending on which level is being displayed, you would see the Master and member lines, plus one of the last three types of lines:

Master (Master)
 Member (Robot)
 Object (Left Arm)
 Property (LeftArm/xfm)
 Field (LeftArm/xfm/translation)

Each entry in the names panel has an associated record button. If a field button is blue, the field value at the current time has not been changed, and you don't need to record it. If a field button is red, you have changed the field, and you should record it in order to save this value in your animation. For field lines, you can force a keyframe to be recorded even if the button is blue.

For property, object, member, and master lines, the record button is either gray or red. Gray indicates that none of the field lines underneath the group-level line has changed. Pressing a gray button has no effect. Red for a group-level line indicates that at least one field line beneath the group is red. When you press a red button on one of these group-level lines, all of the red field-level lines beneath it are recorded, and you will see the group lines return to gray.

Jump to:

- [Master Timeline](#)

Master Timeline



Find it: Click

This page describes the Keyframe Animator master timeline.

The master timeline is the boldface line at the top of the time panel. It indicates the time of the animation, either in frames or in seconds. (Use the *View* menu to switch from frames to seconds.) You can click anywhere in the pink trough above the timeline to specify a new current time.

The red triangle and dotted line, called the *scrub bar*, point to the current time. You can drag the triangle back and forth to preview (or *scrub*) the animation. The blue squares on the master timeline indicate any time that has a keyframe in any of the individual timelines shown below it.

Jump to:

- [Tangent Type](#)

Tangent Type



Find it: Click

The Tangent Type menu allows you to select the type of interpolation that will be used to fill in the values between a given keyframe and the keyframe that follows it. The Keyframe Animator allows you to choose from the following interpolation types:

- Constant
- Linear
- Hermite (the default)
- Split
- Mixed

Constant

The value remains fixed until the next keyframe. No interpolation is performed. This type is useful for things that don't change, such as a blinking neon sign.

Linear

The value changes linearly from the current value to the next keyframe value. This type is useful for light rays bouncing off mirrors at equal angles. For shape animations only, the default interpolation type is Linear.

Hermite

The value passes through the key, but it follows a curve whose tangents are determined by the keys on either end of the segment, and the next one to either side. This type gives a continuous velocity curve. As an example, suppose you have four keys (A, B, C, and D). To find a value on the curve between B and C, the tangent at B is computed as the slope from A to C. The tangent at C is computed as the slope from B to D. And the value of the curve is a spline function of those two tangents and the values at B and C.

Split

This type is similar to Hermite, but it allows the tangent to be discontinuous at this point. For example, if B and C are Split, the tangent at B will be the slope from A to B, not A to C. And the tangent at C will be the slope from C to D, not from B to D.

Mixed

The selected keyframes are of different tangent types.

When you have a keyframe selected, the tangent type button shows its tangent type. If more than one keyframe is selected and the keyframes have different types, the button shows "Mixed."

You can select a keyframe and change its interpolation type by changing the tangent type button to the new type. If there are no keyframes selected, the button label changes to "Default Tangent Type." The value selected will be used for any keyframes that are created or changed.

Jump to:

- [Button Panel](#)

Button Panel



Find it: Click

The button panel contains the following buttons:

- [Disable Interpolation Button](#)
- [Zoom Buttons](#)
- [Create Trigger Button](#)
- [Play and Rewind Buttons](#)

Disable Interpolation Button

The Disable Interpolation button allows you to temporarily disable interpolation. When pressed, it turns red (and the Play button is grayed out). This button is useful when you want to copy a pose, either from a defined keyframe or from an interpolated keyframe. (This technique differs from copy/paste because you can copy in-between poses, not just keyframes.)

Here's how you might use this button:

1. Go to the frame you want to copy FROM.
2. Press the Disable Interpolation button.
3. Go to the frame you want to copy TO.
4. Set the keyframes. You can use the Master record button, or any lower-level buttons to record the keyframes.
5. Press the Disable Interpolation button again so that interpolation resumes. Continue editing.

Zoom Buttons

The *Zoom timeline in* button moves you closer to the window. You see less of the timeline because it is magnified.

The *Default timeline zoom level* button returns the view to its original size.

The *Zoom timeline to fit* button sizes the timeline to fit within the current window.

The *Zoom timeline out* button moves you farther away from the window. The timeline becomes smaller, so you can see more of it.

Create Trigger Button

The Create Trigger button is used to create a sensor in the scene that will start your animation. The button is red until you have explicitly created a sensor for the animation. (If nothing is selected, it is grayed out.) Initially, a default touch sensor is automatically created so that you can preview the animation in Cosmo Player. Once you create your own sensor, the default touch sensor is removed.

When you press the button, the sensor panel appears. Select a sensor and press *OK*. Current choices for the sensor are

- Collision Node - trigger the animation when the user collides with a particular object
- World Entry Script - trigger the animation when the scene is first loaded
- Proximity Sensor - trigger when the user is within a certain distance of an object
- Touch Sensor - trigger when the user clicks a particular object
- Viewpoint Binding - trigger when a viewpoint is bound (locked) to the camera
- Visibility Sensor - trigger when a particular object or group of objects is visible

Once you have created a trigger, the button turns black. You can press the button again to create additional triggers.

Play and Rewind Buttons

This set of buttons is similar to the set of buttons on a VCR. The middle button is Stop. The button on the far right is Play, and the button on the far left is Rewind to the start. The second button from the left steps back one frame. The second button from the right steps forward one frame.

Jump to:

- [Keyframe Animator Quick Reference](http://oldsite.vislab.usyd.edu.au/users/manuals/internet/cosmoworlds/Reference/keybut.htm)

Deleting, Copying, and Pasting Polygons

You can delete, copy, and paste polygons, but not individual points or edges. You can copy polygons from one PEP object and paste them into another PEP object.

Set up:

1. Make sure you are in the [PEP editor](#).
2. [Select polygons](#) that you want to delete.



To delete selected polygons: Press this button, or use *Delete* or *Ctrl-x*.

To copy selected polygons: Use *Ctrl-c*.

To paste copied polygons on top of copied selection: Use *Ctrl-v*.

Jump to:

- [PEP Modeling](#)
- [Tools to Use: Creating and Editing Models](#)

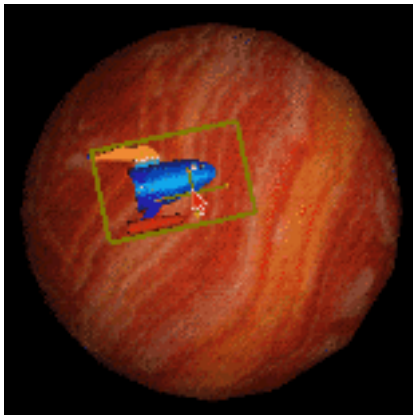
Creating and Animating a Rocket

Level: Beginner to Intermediate

Goal: This tutorial gives you hands-on experience with Cosmo Worlds' most-commonly used features, tools, and editors.

Assumptions: This tutorial is divided into four parts. Each section, for example, *Part 1: Making a Planet*, assumes you have followed the steps from beginning to end for that part and have followed the *Jump to* links for background information. Use the Back button on Netscape's menu bar to return to the tutorial after following such links.

Finished product:



This tutorial consists of the following sections:

- [Part 1: Making a Planet](#)
- [Part 2: Creating a Rocket](#)
- [Part 3: Animating the Rocket](#)
- [Part 4: Previewing the World](#)

Use Undo and Redo to fix mistakes:

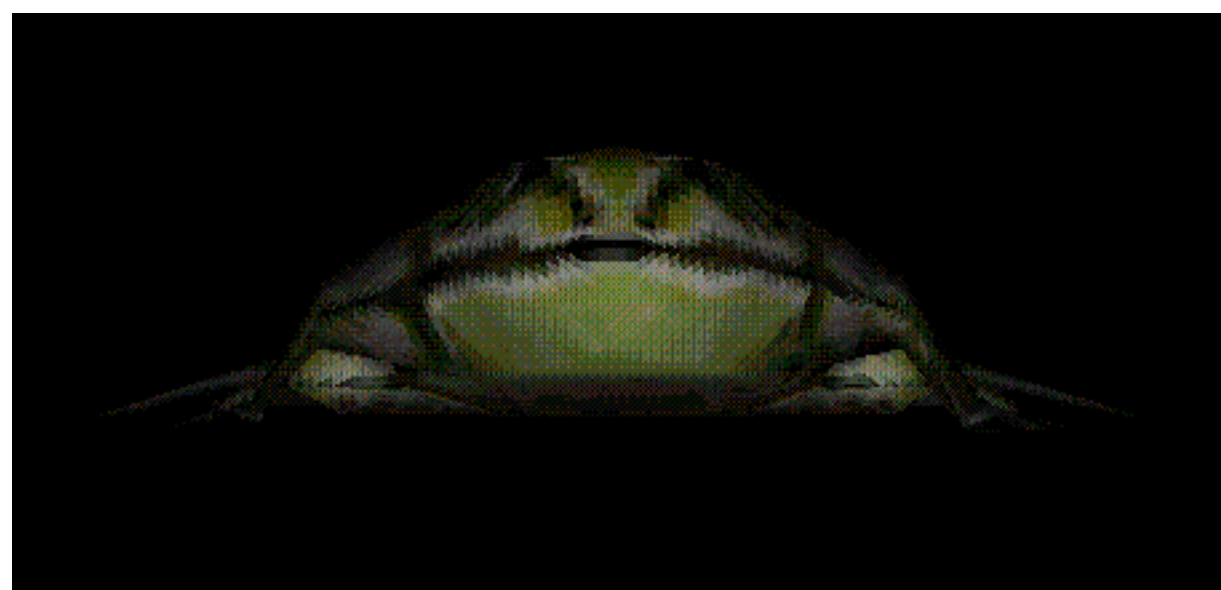
- Undo = *Ctrl+Z*
- Redo = *Shift+Ctrl+Z*

[Part 1: Making a Planet](#) ➡

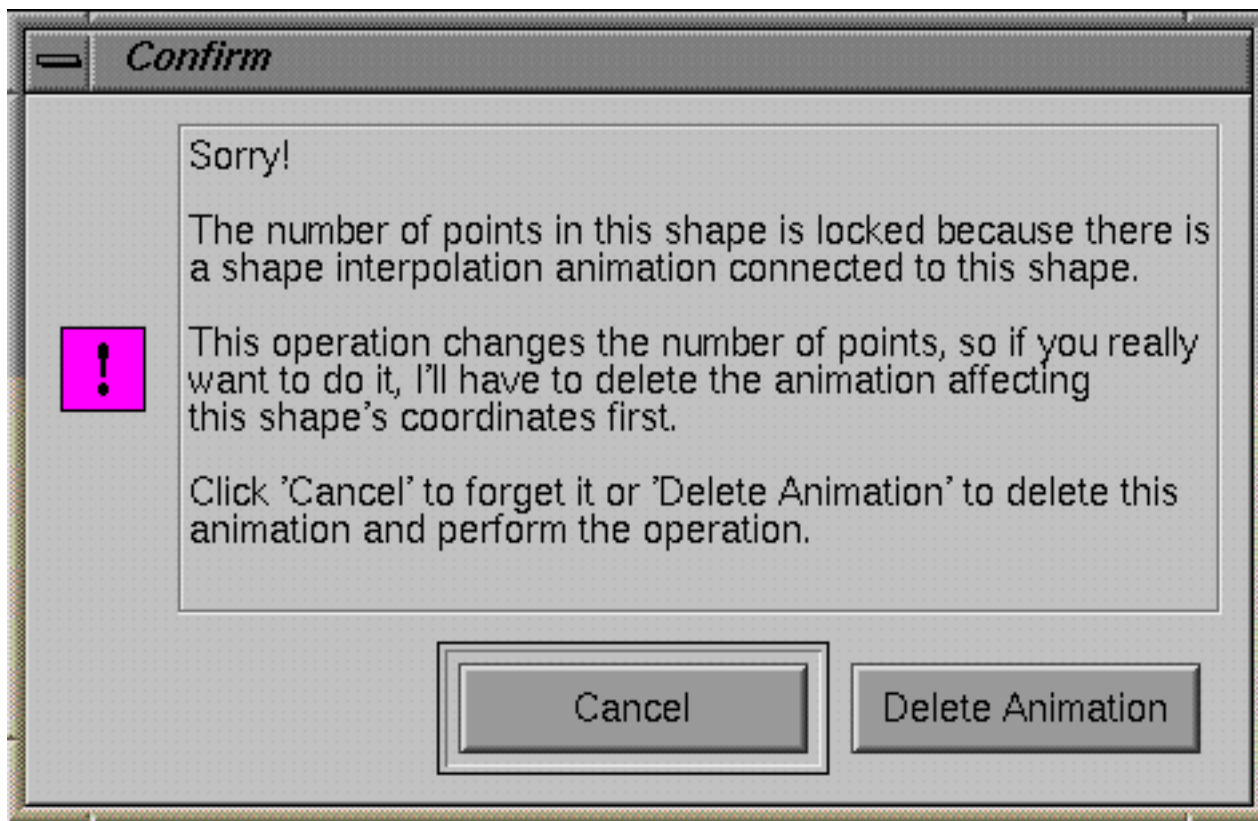
Try It! Animating a Shape

Animated shapes add realism to a scene: you can watch the throat of a frog expand and contract as he breathes, and you can see a bouncing rubber ball flatten slightly when it hits the ground. The technique for animating a shape is similar to the technique for animating the movements of an object: you move the time marker in the keyframe animator to the appropriate time, change the points, edges, or polygons of a PEP object, and then record the keyframe.

[Click here](#) to preview an animation similar to the one you're about to create.



If a PEP object has a shape animation associated with it, you won't be able to use any of the PEP editing tools to split polygons or delete polygons from the shape. If you try to alter the number of points in an animated shape, the following message appears:



Be sure to make all changes to the topology of your shape before you begin recording the animation.


Reminder: Use *Edit > Undo* (*Ctrl + Z*) and *Edit > Redo* (*Shift + Ctrl + Z*) in the main window to undo and redo changes made in the Keyframe Animator as well as changes made in the main window. The Keyframe Animator uses the same undo buffer as the main application.

Animating a Simple Shape

Follow these steps to set up the animation:

1. In Cosmo Worlds, open the sample file */usr/share/Insight/library/SGL_bookshelves/Help/books/CosmoWorlds_UG/Models/frog.wrl*

If you were going to edit a primitive object such as a sphere, you'd need to click the *Convert to*

PEP Object button: 

The frog is already a PEP object, so the PEP hammer is grayed out.

2. Click the *PEP Editor* button: 

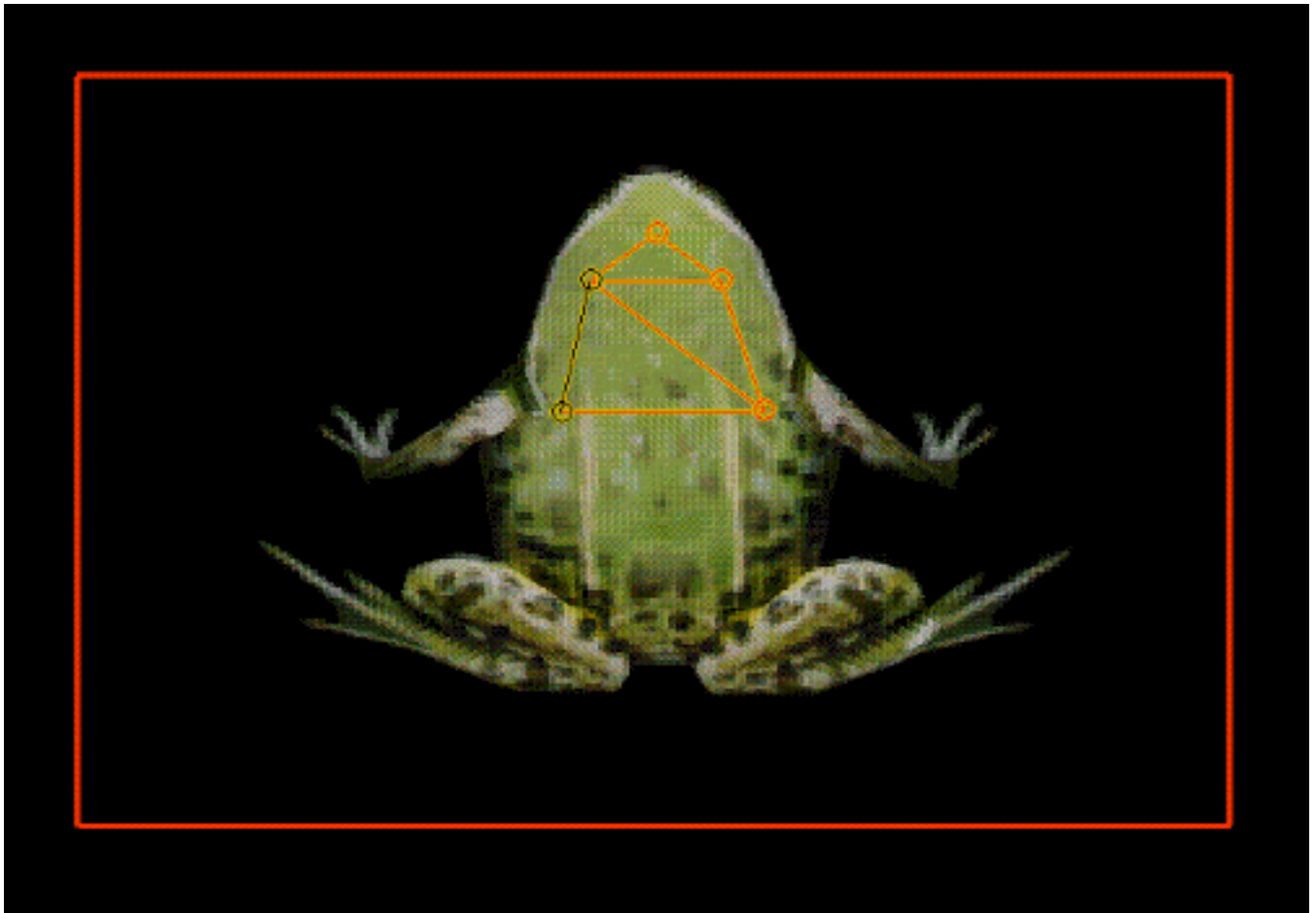
A dialog box appears asking you to select a child. Dismiss the box and then click the *Select Child* down arrow to select the bottom half of the frog's body. Click the PEP Editor button; an orange box appears to indicate PEP editing mode.



3. Start the Keyframe Animator by clicking its button:
4. Choose *Animation > New animation*. Then choose *Animation > Change name* and type in a name for your animation: *myfrog*.
5. Add the frog as a member of the animation by choosing *Animation > Add member*.
6. Choose *Animation > Set duration* and set the duration to 10 frames.

Now you're ready to animate the shape:

1. Move the current time marker (the red triangle) in the Keyframe Animator to frame 5.
2. In the main window, *Shift*-click to [select the polygons](#) that form the throat of the frog, as shown in the figure below.



3. Click-drag the selection to expand the frog's throat. (To get a better angle, hold down the *Alt* key to switch temporarily to the Examiner Viewer so that you can rotate the frog.)

If you press the *Ctrl* key when you click-drag, you can pull out the vertices in a direction perpendicular to the surface of the frog's throat.

4. In the Keyframe Animator, click the master Record button to record the keyframe.

5. Move the current time marker to frame 10.
6. In the main window, click-drag the frog's throat to its original position (or simply [copy and paste the keys](#) from frame 0 to frame 10).
7. Click the master Record button to record the keyframe.
8. In the Keyframe Animator, click the *Play* arrow to preview the animation.

Jump to:

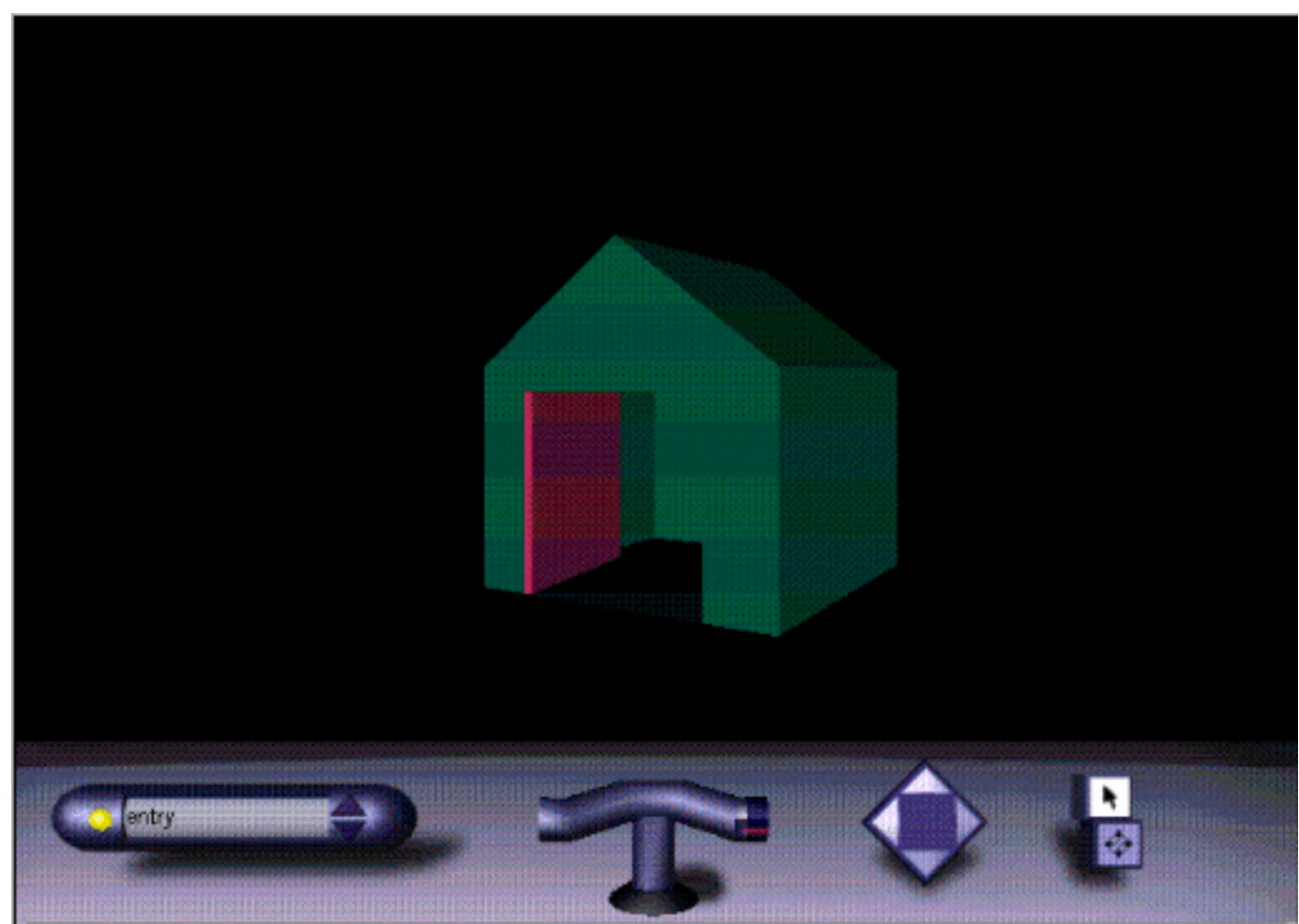
- [Keyframe Animator Quick Reference](#)
- [Try It: Keyframe Animator!](#)

Try It! Creating an Event Sound

An *event* sound is associated with something that happens in a scene--for example, a door banging as it closes or a pot falling off a table and crashing on the floor. An event sound usually requires a script to synchronize the sound with the accompanying action.

The following file contains an example of an event sound and a simple script:

`/usr/share/Insight/library/SGI_bookshelves/Help/books/CosmoWorlds_UG/Models/doorSample.wrl`



[Click here](#) to preview the file. Click the door to make it close. Notice how the sound occurs at the end of the door's swing cycle, just when it is in the closed position. The touch sensor's start time is routed to a script, which adds a delay of 1.8 seconds to produce the start time for the door sound. The door animation is also triggered by a touch sensor.

To view this file and examine the routes in the Outline Editor, [click here](#). (The blue routes are for the door animation. The black routes are for the sound.)

Steps for Creating an Event Sound

Follow these steps to create an event sound (refer to the example file *doorSample.wrl* to help you get started):

1. In Cosmo Worlds, open the file */usr/share/Insight/library/SGI_bookshelves/Help/books/CosmoWorlds_UG/Models/door.wrl*

This file contains the house and door model as well as a keyframe animation that opens and closes the door.

2. Click the Sound Editor button on the *Editors* palette.
3. Select the door, which is the object associated with the event.
4. In the Sound Editor, click the *Create Local* button. (If the button is grayed out, choose *Edit > Create Parent Group* to enable the *Create Local* button.)
5. Click the *Browse* button and choose a sound file to add to the scene. Click *OK*.
6. The sound is named *Sound1* by default. Type a new name for your sound in the text field--for example, *doorSlam*.
7. Use the green sound icon to position the sound at the site of the slamming door. The sound icon can be positioned and rotated in the same way as the standard manipulator. (If the icon is not visible, choose *View > Show/Hide Iconic Objects > Show Sounds*.)
8. Type new range values into the text fields for *Max Front*, *Max Back*, *Min Front*, and *Min Back*. To check the range of the sound, click the *Show Range* box. A wireframe sphere indicates the range of the sound.

In the door example, notice how the sound range is mostly in front of the door. Also, because the sound range is not spherical, the range values need to be typed into the fields (that is, you can't simply use the thumbwheel).

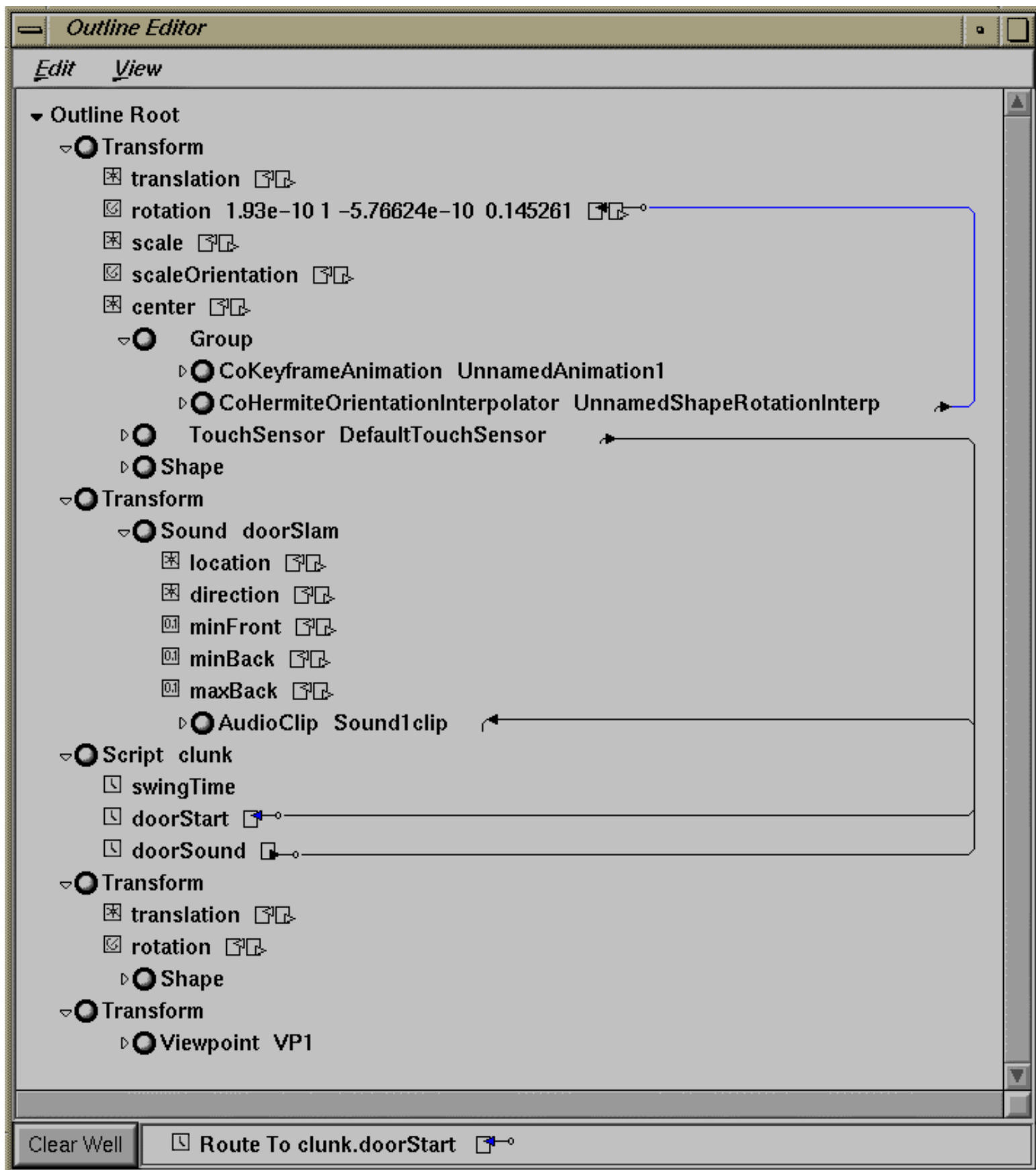
Tip: If you want to create a sound range with a shape not directly supported by the Sound Editor, you can choose *Edit > Create Parent Group* to create a Transform grouping node above the sound node. Then, in the Outline Editor, you can scale the Sound to create a different shape, such as a flattened ellipsoid.

9. Use the default settings for spatialization (on) and looping (off).
10. Click the *Create Trigger to Activate Clip* button to create a trigger for the sound. Event sounds are typically triggered by a touch sensor, a collision node, or a proximity sensor. (The door example uses a touch sensor.)
11. If you want to synchronize the sound with something other than the start time of the sensor, write a script that builds in the proper timing of the sound with the event. The door example uses a [script](#) to delay the noise until the door is almost closed.

12. Click the arrow button to preview the sound.

Jump to:

- [Sound Editor Quick Reference](#)
- [Recording Sounds](#)



Recording Sounds

Use one of the following tools to help you record and edit sounds:

- **Media Recorder** - for recording a simple sound
- **Sound Editor** (located on the Media Tools page of the icon catalog) - for simple editing of single-track audio files
- **Sound Track** - for composing and editing multitrack compositions

You may also want to use sounds from a sound effects library, available from many commercial sources.

For best performance, record your sound using a sample rate of 22.05 Hz. (This is half the sample rate of a standard CD.) Use the same sample rate for all sounds in your scene.

Jump to: [Sound Editor Quick Reference](#)

Cosmo™ Worlds Help

Document Number 007-3312-002

Cosmo Worlds Version 1.0.2

CONTRIBUTORS

Written by Julie Brodeur and Josie Wernecke.

Background, examples, good ideas, and support by the Cosmo Worlds team.

Product Design and Engineering: Adam Feder, Alain Dumesny, Chris Fouts, Craig Kolb, Curtis Beeson, Dave Immel, David Story, Geoff Brown, Jackie Neider, Jeff White, Jim Miller, Kurt Schaefer, Michael Natkin, Paul Isaacs, Paul Strauss, Rich Gossweiler, Rick Pasetto, Rob Myers.

Quality Assurance: Allen Henry, Bill Day, Ina Canlas, Kristi Taylor, Sarah Liberman. *Usability:* Deb Galdes. *Bleeding-edge users:* David Frerichs, Delle Maxwell, Kevin Hartz, Rob Lewis, Sam Chen.

Additional good ideas and support by Baron Roberts and Doug O'Morain.

© Copyright 1996, 1997 Silicon Graphics, Inc. - All Rights Reserved

The contents of this document may not be copied or duplicated in any form, in whole or in part, without the prior written permission of Silicon Graphics, Inc.

RESTRICTED RIGHTS LEGEND

Use, duplication, or disclosure of the technical data contained in this document by the Government is subject to restrictions as set forth in subdivision (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 52.227-7013 and/or in similar or successor clauses in the FAR, or in the DOD or NASA FAR Supplement. Unpublished rights reserved under the Copyright Laws of the United States. Contractor/manufacturer is Silicon Graphics, Inc., 2011 N. Shoreline Blvd., Mountain View, CA 94039-7311.

Silicon Graphics is a registered trademark and Cosmo is a trademark of Silicon Graphics, Inc. Netscape is a trademark of Netscape Communications Corporation.

QBullets are a service trademark of Michael Herrick/Matterform Media.🌐

Contents

Tasks

- [Introductory material](#)
 - [Overview](#)
 - [Copyright](#)
- [Getting started](#)
- [Tutorials](#)
- [Opening and saving a scene](#)
 - [Saving objects from a scene](#)
- [Viewing objects and scenes](#)
 - [Examining an object \(rotating, zooming, panning\)](#)
 - [Walking through a scene](#)
 - [Setting a home view](#)
 - [Viewing all models in a scene](#)
 - [Seeking to a point or object](#)
 - [Setting viewer preferences](#)
 - [Defining viewpoints](#)
 - [Global vs. local](#)
- [Importing objects](#)
 - [Importing geometry](#)
 - [Importing an image](#)
 - [Importing a sound](#)
 - [Importing objects as inlines](#)
- [Manipulating objects](#)
 - [Placing shapes when importing, pasting, and creating](#)
 - [Selecting and grouping objects](#)
 - [Creating group hierarchies](#)
 - [Cloning an object](#)
 - [Resizing, moving, and rotating an object](#)
 - [Constraining movement and rotation to a single plane](#)
 - [Aligning two objects](#)
 - [Specifying an object's position in the scene](#)
- [Creating basic shapes and text](#)
 - [Making tube-shaped models](#)
 - [Creating text](#)
- [PEP modeling: editing points, edges, and polygons](#)
 - [Viewing PEP Objects](#)
 - [Selecting PEPs](#)

- [Translating PEPs](#)
- [Aligning PEPs](#)
- [Rotating and scaling PEPs](#)
- [Deleting, copying, and pasting polygons](#)
- [Merging, combining, or joining PEPs](#)
- [Chipping off and forming new polygons](#)
- [Splitting and cutting polygons](#)
- [Extruding polygons and edges](#)
- [Copying polygons and rebuilding models](#)
- [Mirroring and reordering polygons](#)
- [Changing an object's appearance](#)
 - [Changing the color of a model](#)
 - [Changing the material of an object](#)
 - [Creating your own material](#)
 - [Applying a texture to a model](#)
 - [Editing a texture](#)
 - [Fine-tuning a texture](#)
 - [Creating a new texture palette](#)
 - [Adjusting normals](#)
- [Animating 3D objects](#)
 - [Animating 3D models](#)
 - [Selecting, copying, and pasting keyframes](#)
 - [Animating viewpoints](#)
 - [Try it! Animating a shape](#)
 - [Creating a script:overview](#)
 - [Scripting basics](#)
 - [Steps for creating a script](#)
 - [Try it! Case study for creating a script](#)
 - [Debugging a script](#)
 - [Tips for creating scripts](#)
 - [Advanced: creating a script from scratch](#)
 - [Creating a switch](#)
 - [Creating a trigger](#)
- [Adding special objects to a scene](#)
 - [Adding navigation information](#)
 - [Adding lights](#)
 - [Tips for efficient lighting](#)
 - [Light types: what's the difference?](#)
 - [More about spotlights](#)
 - [Try It! Experimenting with lighting effects](#)
 - [Adding sounds](#)
 - [Try it! Creating an ambient sound](#)
 - [Try it! Creating an event sound](#)

- [Recording sounds](#)
- [Adding links \(anchors\)](#)
- [Adding billboards](#)
- [Using the outline editor](#)
 - [Creating routes](#)
 - [Cloned nodes in scripts and prototypes](#)
- [Optimizing the scene](#)
 - [Checking the polygon count](#)
 - [Reducing polygon count](#)
 - [Building efficient polygons](#)
 - [Using inlined objects](#)
 - [Using multiple levels of detail](#)
 - [Controlling collision detection](#)
- [Packaging the scene](#)
 - [Steps for packaging](#)
 - [Try it! Packaging a sample world](#)
 - [Links followed during discovery](#)
 - [Possible packaging errors](#)
- [Customizing the environment](#)
 - [Adding a helper application](#)
 - [Helper application database file](#)
 - [Changing the background color](#)

Reference

The Cosmo Worlds Window

- [A quick look at the user interface](#)
(menus and palettes)
- [Keys and shortcuts](#)
- [Viewers](#)
 - [Examiner Viewer](#)
 - [Walk Viewer](#)
 - [Plane Viewer](#)
 - [Viewer Popup Menu](#)
 - [Viewer Menu Preferences](#)

Cosmo Worlds Tools and Utilities

- Creators
 - [Text Editor](#)
 - [Extrusion Editor](#)

- Editors

- [Outline Editor](#)
- [NavigationInfo Editor](#)
- [Sound Editor](#)
- [Viewpoint Editor](#)
- [Polygon Copier](#)
- [Polygon Reduction Editor](#)

- [Media Tools](#)

Contents depends on system configuration; possibilities include:

- ImageWorks Plug-in Editor
- Alias/Wavefront Plug-in Editor
- Photoshop Plug-in Editor

- Action Tools

- [Script Editor](#)
- [Keyframe Animator](#)
- [Link Editor](#)
- [Inline Editor](#)
- [Switch Editor](#)
- [Level of Detail Editor](#)
- [Billboard Editor](#)
- [Collision Editor](#)

- [Layout Tools](#)

- [Snap Target and Snap Source](#)
- [Precision placement panel](#)
- [PEP Alignment](#)
- [PEP distance snap options](#)
- [PEP angle snap option](#)

- [PEP Editing Tools](#)

- [Selection](#)
- [PEP Editor](#)
- [Convert to PEP](#)
- [PEP Rotation / Scaling](#)
- [Cut Polygons](#)
- [Merge PEP Objects](#)
- [Join Selected Polygons](#)
- [Chip Off and Split Selected Polygons](#)
- [Extrude Selected Polygons or Edges](#)

- Looks Tools

- [Material Palette](#)
- [Normal Doctor](#)
- [Material Editor](#)
- [Texture Editor](#)
- [Color Per Vertex Editor](#)
- [PEP Texture Applicator](#)
- [Light Editor](#)

- [CosmoPackage](#)
 - [Root documents](#)
 - [VRML preferences](#)
 - [Directory index](#)
 - [Mappings](#)
 - [Trusted references](#)
 - [Inspecting documents](#)
- [Glossary](#)

Changing an Object's Appearance

Using the editors found on the *Looks* palette, you can alter an object's appearance in several ways. Learn about:

- [Changing the color of a model](#)
- [Changing the material of an object](#)
- [Creating your own material](#)
- [Applying a texture to a model](#)
- [Editing a texture](#)
- [Fine-tuning a texture on a model](#)
- [Creating a new texture palette](#)
- [Adjusting normals](#)

Note: The Material Palette and Texture Editor use Showcase's default palettes. If you do not have Showcase installed, the editors will still work, but you won't have access to pre-made palettes.

Adding Special Objects to a Scene

You can add a number of special objects to your scene:

Lights

Spot lights, point lights, and directional lights can add exciting visual effects to a scene.

Sounds

The added realism of crickets chirping, water splashing, and pipes clanking can enhance your scene too.

Links

Click on a link to an HTML document or enter a new VRML world.

Billboards

Billboards are objects that rotate around a specified axis and always face the viewer. They're a useful optimization technique.

Collision Detection Groupings

You can turn off collision detection or provide a proxy (substitute object) for testing collisions. Collision detection groupings are also useful for optimization.

Navigation Info Objects

You can include objects that store information about the scene, such as the initial viewer type, whether to turn on a headlight, the size of the avatar, and the height of objects over which it can step.

Creating an Ambient Sound

An *ambient* sound plays at the same volume everywhere in the scene. Crickets in a field, birds chirping in a forest, and the whirring and hammering of factory machines are examples of ambient sounds.

Follow these steps to create an ambient sound:

1. Click the Sound Editor button on the *Editors* palette.
2. Click the *Create Global* button.
3. Click the *Browse* button and choose a file to add to the scene. For example, try browsing for a cricket sound. Click *OK*.
4. The sound is named *Sound1* by default. Type a new name for your sound in the text field--for example, *crickets*.
5. Use the *Resize Range* thumbwheel to increase the range of the sound so that it encompasses the entire scene. To check the range of the sound, click the *Show Range* box. A wireframe sphere indicates the range of the sound.

The thumbwheel extends the range up to 100 meters. If you have a large world, you may need to type the numbers into the *Max Range* text fields.

6. Click the *Spatialize* box to remove the check mark and turn spatialization off.
7. Click the *Loop* box to make the sound play continuously.
8. Click the *Create Trigger to Activate Clip* button to create a trigger for the sound. Ambient sounds are usually triggered by the *World Entry Script* trigger.
9. Click the arrow button to preview the sound.

Jump to: [Sound Editor Quick Reference](#)

Customizing the Environment

You can adjust your toolbar, manipulate palettes, and add plug-ins. Choose *File > Save Layout* to save the current toolbar and palettes.

Learn about:

- [Modifying the Toolbar](#)
- [Manipulating Palettes](#)
- [Adding Plug-ins](#)

Changing the Background

The Nodes Gallery (*usr/share/data/vrml*) contains two Background nodes that you can add to your scene. The files are

Background.wrl

This file adds a solid blue background to your scene.

BackgroundSunny.wrl

This file adds a ground plane that fades to a sky plane with a sun in it.

Add the background to your scene either by dragging and dropping the file icon from the desktop or by using the *File > Import* menu. (**Note:** you may receive a warning message, but you can ignore it and continue.)

Currently, you can't change the number of ground angles or sky angles in the Background node, but you can modify their values in the Outline Editor. To change the number of angles, edit the original file containing the Background node before you import it into your scene.

Contents

Tasks

- [Introductory material](#)
 - [Overview](#)
 - [Copyright](#)
- [Getting started](#)
- [Tutorials](#)
- [Opening and saving a scene](#)
 - [Saving objects from a scene](#)
- [Viewing objects and scenes](#)
 - [Examining an object \(rotating, zooming, panning\)](#)
 - [Walking through a scene](#)
 - [Setting a home view](#)
 - [Viewing all models in a scene](#)
 - [Seeking to a point or object](#)
 - [Setting viewer preferences](#)
 - [Defining viewpoints](#)
 - [Global vs. local](#)
- [Importing objects](#)
 - [Importing geometry](#)
 - [Importing an image](#)
 - [Importing a sound](#)
 - [Importing objects as inlines](#)
- [Manipulating objects](#)
 - [Placing shapes when importing, pasting, and creating](#)
 - [Selecting and grouping objects](#)
 - [Creating group hierarchies](#)
 - [Cloning an object](#)
 - [Resizing, moving, and rotating an object](#)
 - [Constraining movement and rotation to a single plane](#)
 - [Aligning two objects](#)
 - [Specifying an object's position in the scene](#)
- [Creating basic shapes and text](#)
 - [Making tube-shaped models](#)
 - [Creating text](#)
- [PEP modeling: editing points, edges, and polygons](#)
 - [Viewing PEP Objects](#)
 - [Selecting PEPs](#)
 - [Translating PEPs](#)

- [Aligning PEPs](#)
 - [Rotating and scaling PEPs](#)
 - [Deleting, copying, and pasting polygons](#)
 - [Merging, combining, or joining PEPs](#)
 - [Chipping off and forming new polygons](#)
 - [Splitting and cutting polygons](#)
 - [Extruding polygons and edges](#)
 - [Copying polygons and rebuilding models](#)
 - [Mirroring and reordering polygons](#)
- [Changing an object's appearance](#)
 - [Changing the color of a model](#)
 - [Changing the material of an object](#)
 - [Creating your own material](#)
 - [Applying a texture to a model](#)
 - [Editing a texture](#)
 - [Fine-tuning a texture](#)
 - [Creating a new texture palette](#)
 - [Adjusting normals](#)
- [Animating 3D objects](#)
 - [Animating 3D models](#)
 - [Selecting, copying, and pasting keyframes](#)
 - [Animating viewpoints](#)
 - [Try it! Animating a shape](#)
 - [Creating a script:overview](#)
 - [Scripting basics](#)
 - [Steps for creating a script](#)
 - [Try it! Case study for creating a script](#)
 - [Debugging a script](#)
 - [Tips for creating scripts](#)
 - [Advanced: creating a script from scratch](#)
 - [Creating a switch](#)
 - [Creating a trigger](#)
- [Adding special objects to a scene](#)
 - [Adding navigation information](#)
 - [Adding lights](#)
 - [Tips for efficient lighting](#)
 - [Light types: what's the difference?](#)
 - [More about spotlights](#)
 - [Try It! Experimenting with lighting effects](#)
 - [Adding sounds](#)
 - [Try it! Creating an ambient sound](#)
 - [Try it! Creating an event sound](#)
 - [Recording sounds](#)
 - [Adding links \(anchors\)](#)

- [Adding billboards](#)
- [Using the outline editor](#)
 - [Creating routes](#)
 - [Cloned nodes in scripts and prototypes](#)
- [Optimizing the scene](#)
 - [Checking the polygon count](#)
 - [Reducing polygon count](#)
 - [Building efficient polygons](#)
 - [Using inlined objects](#)
 - [Using multiple levels of detail](#)
 - [Controlling collision detection](#)
- [Packaging the scene](#)
 - [Steps for packaging](#)
 - [Try it! Packaging a sample world](#)
 - [Links followed during discovery](#)
 - [Possible packaging errors](#)
- [Customizing the environment](#)
 - [Adding a helper application](#)
 - [Helper application database file](#)
 - [Changing the background color](#)

Reference

The Cosmo Worlds Window

- [A quick look at the user interface](#)
(menus and palettes)
- [Keys and shortcuts](#)
- [Viewers](#)
 - [Examiner Viewer](#)
 - [Walk Viewer](#)
 - [Plane Viewer](#)
 - [Viewer Popup Menu](#)
 - [Viewer Menu Preferences](#)

Cosmo Worlds Tools and Utilities

- Creators
 - [Text Editor](#)
 - [Extrusion Editor](#)
- Editors
 - [Outline Editor](#)
 - [NavigationInfo Editor](#)

- [Sound Editor](#)
- [Viewpoint Editor](#)
- [Polygon Copier](#)
- [Polygon Reduction Editor](#)

- [Media Tools](#)

Contents depends on system configuration; possibilities include:

- ImageWorks Plug-in Editor
- Alias/Wavefront Plug-in Editor
- Photoshop Plug-in Editor

- Action Tools

- [Script Editor](#)
- [Keyframe Animator](#)
- [Link Editor](#)
- [Inline Editor](#)
- [Switch Editor](#)
- [Level of Detail Editor](#)
- [Billboard Editor](#)
- [Collision Editor](#)

- [Layout Tools](#)

- [Snap Target and Snap Source](#)
- [Precision placement panel](#)
- [PEP Alignment](#)
- [PEP distance snap options](#)
- [PEP angle snap option](#)

- [PEP Editing Tools](#)

- [Selection](#)
- [PEP Editor](#)
- [Convert to PEP](#)
- [PEP Rotation / Scaling](#)
- [Cut Polygons](#)
- [Merge PEP Objects](#)
- [Join Selected Polygons](#)
- [Chip Off and Split Selected Polygons](#)
- [Extrude Selected Polygons or Edges](#)

- Looks Tools

- [Material Palette](#)
- [Normal Doctor](#)
- [Material Editor](#)
- [Texture Editor](#)
- [Color Per Vertex Editor](#)
- [PEP Texture Applicator](#)
- [Light Editor](#)

- [CosmoPackage](#)

- [Root documents](#)
- [VRML preferences](#)
- [Directory index](#)

- [Mappings](#)
- [Trusted references](#)
- [Inspecting documents](#)
- [Glossary](#)

[Overview](#) | [Tasks](#) | [Reference](#) | [LargeHelp](#)
[resizing the help window](#)