Tutorials:
Special Effects

Autodesk® 3ds Max®

2010

Autodesk®

Autodesk® 3ds® Max 2010 Software

The following are registered trademarks or trademarks of Autodesk, Inc., in the USA and other countries: 3DEC (design/logo), 3December, 3December.com, 3ds Max, ADI, Alias, Alias (swirl design/logo), AliasStudio, Alias|Wavefront (design/logo), ATC, AUGI, AutoCAD, AutoCAD Learning Assistance, AutoCAD LT, AutoCAD Simulator, AutoCAD SQL Extension, AutoCAD SQL Interface, Autodesk, Autodesk Envision, Autodesk Insight, Autodesk Intent, Autodesk Inventor, Autodesk Map, Autodesk MapGuide, Autodesk Streamline, AutoLISP, AutoSnap, AutoSketch, AutoTrack, Backdraft, Built with ObjectARX (logo), Burn, Buzzsaw, CAiCE, Can You Imagine, Character Studio, Cinestream, Civil 3D, Cleaner, Cleaner Central, ClearScale, Colour Warper, Combustion, Communication Specification, Constructware, Content Explorer, Create>what's>Next> (design/logo), Dancing Baby (image), DesignCenter, Design Doctor, Designer's Toolkit, DesignKids, DesignProf, DesignServer, DesignStudio, Design|Studio (design/logo), Design Web Format, Discreet, DWF, DWG, DWG (logo), DWG Extreme, DWG TrueConvert, DWG TrueView, DXF, Ecotect, Exposure, Extending the Design Team, Face Robot, FBX, Filmbox, Fire, Flame, Flint, FMDesktop, Freewheel, Frost, GDX Driver, Gmax, Green Building Studio, Heads-up Design, Heidi, HumanIK, IDEA Server, i-drop, ImageModeler, iMOUT, Incinerator, Inferno, Inventor, Inventor LT, Kaydara, Kaydara (design/logo), Kynapse, Kynogon, LandXplorer, LocationLogic, Lustre, Matchmover, Maya, Mechanical Desktop, Moonbox, MotionBuilder, Movimento, Mudbox, NavisWorks, ObjectARX, ObjectDBX, Open Reality, Opticore, Opticore Opus, PolarSnap, PortfolioWall, Powered with Autodesk Technology, Productstream, ProjectPoint, ProMaterials, RasterDWG, Reactor, RealDWG, Real-time Roto, REALVIZ, Recognize, Render Queue, Retimer,Reveal, Revit, Showcase, ShowMotion, SketchBook, Smoke, Softimage, Softimage|XSI (design/logo), SteeringWheels, Stitcher, Stone, StudioTools, Topobase, Toxik, TrustedDWG, ViewCube, Visual, Visual Construction, Visual Drainage, Visual Landscape, Visual Survey, Visual Toolbox, Visual LISP, Voice Reality, Volo, Vtour, Wire, Wiretap, WiretapCentral, XSI, and XSI (design/logo).

Trademarks

The following are registered trademarks or trademarks of Autodesk Canada Co. in the USA and/or Canada and other countries: Backburner, Multi-Master Editing, River, and Sparks.
The following are registered trademarks or trademarks of Moldflow Corp. in the USA and/or other countries: Moldflow MPA, MPA (design/logo), Moldflow Plastics Advisers, MPI, MPI (design/logo), Moldflow Plastics Insight, MPX, MPX (design/logo), Moldflow Plastics Xpert.
clothfx™ is a trademark of Size8 Software, Inc. Havok.com™ is a trademark or registered trademark of Havok.com Inc. or its licensors. Intel is a registered trademark of Intel Corporation. mental ray is a registered trademark of mental images GmbH licensed for use by Autodesk, Inc. All other brand names, product names or trademarks belong to their respective holders.

Disclaimer

Special Effects

9

This section covers a range of effects: liquid flowing through pipes, lens effects for a camera image, even tumbling down a set of stairs! You will learn how to use particle systems, add lens effects, as well as create many simulations using the reactor toolset.

Features Covered in This Section

- Creating particle systems.
- Using different lens effects.
- Creating rigid body collections.
- Creating simulations with reactor.
- Setting physical properties for objects in simulation.
- Using Hinge and Rag Doll constraints.

Working with Particle Flow

This section contains tutorials that step you through the basics of how to use the Particle Flow particle system in 3ds Max.

For more tutorials on creating special effects with Particle Flow such as water splashing, mist blowing, the explosive impact of an asteroid into a planet surface, guided missiles trailing smoke and blowing up targets, and more, download the 3ds Max tutorials from *this page*.

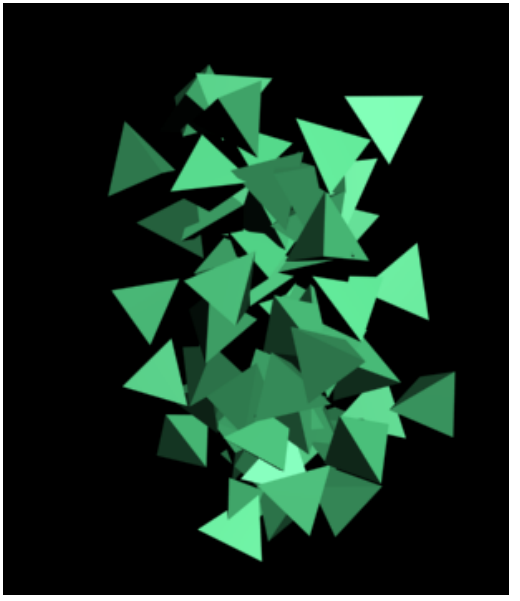
Features Covered in This Section

- Introduction to Particle Flow and the powerful Particle View editor

- Changing particle shape and color based on test results
- Using BlobMesh with a particle system to simulate viscous fluids
- Using instanced geometry to create particles

Introductory Tutorial

The best way to understand Particle Flow is by using it. This tutorial will acquaint you with some of the basic methods of working with Particle Flow.



Skill level: Beginner

Time to complete: 30 minutes

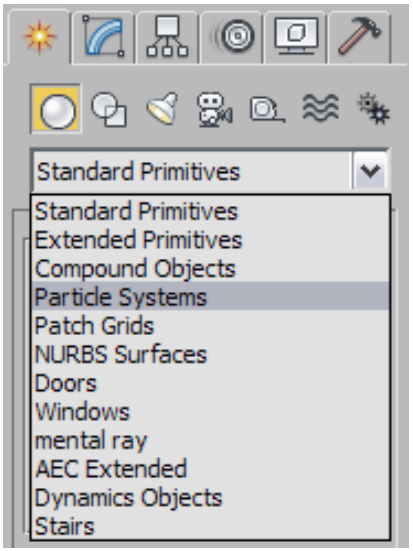
In this tutorial, you will learn how to:

- Create a particle system
- Set up an event that triggers particle activity
- View particle flow animation

Create the Particle Flow system:

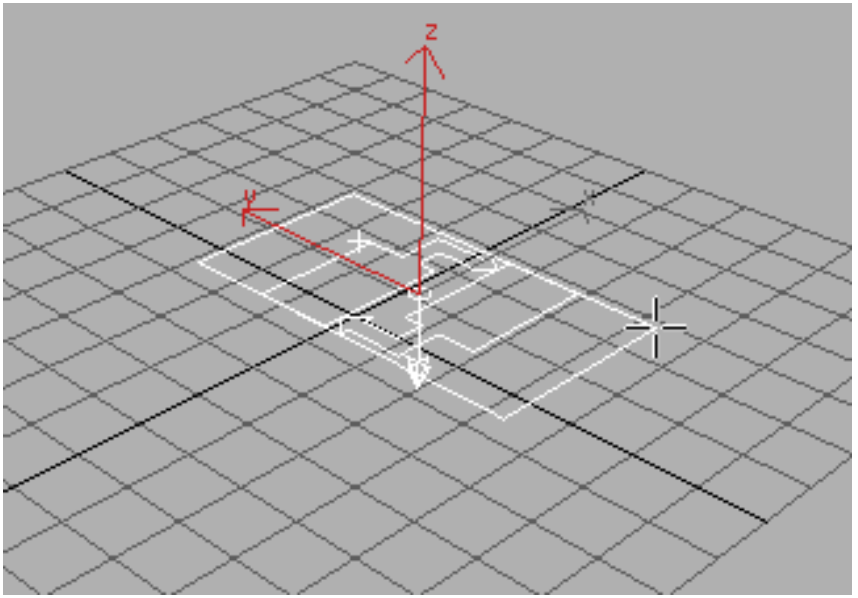
1  Start 3ds Max or, from the Application menu, choose Reset.

2 On the Create panel > Geometry category, click the drop-down list and choose Particle Systems.



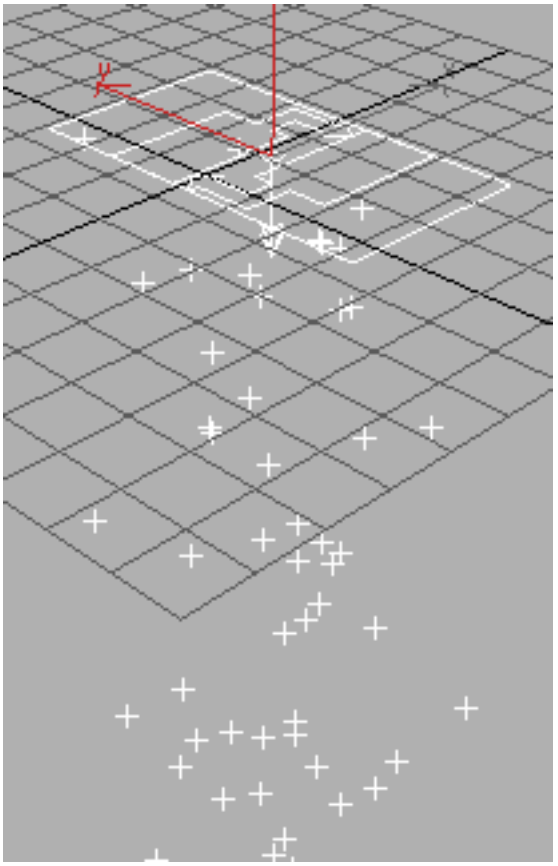
3 On the Object Type rollout, click PF Source.

4 In the Perspective viewport, drag out a rectangle.



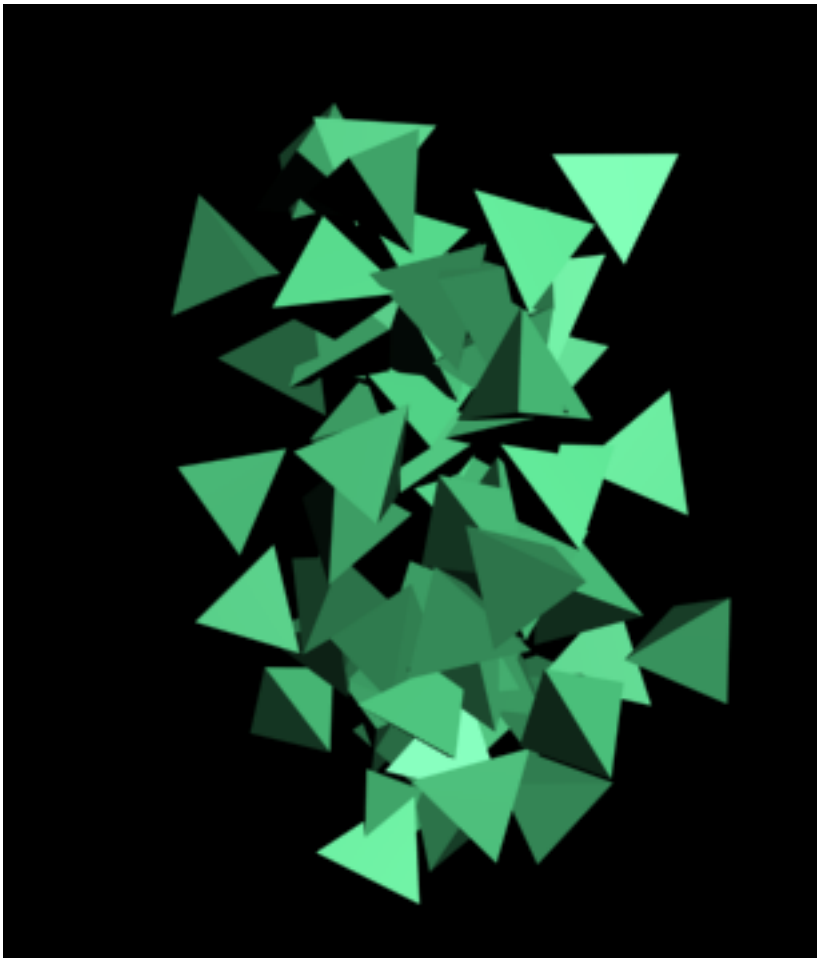
This is the Particle Flow icon, or source, named PF Source 01. By default, it acts as an emitter, but you can also use any other object as an emitter.

5 Drag the time slider.



By default, the icon emits particles downward from its entire surface. In the viewport, the particles appear as ticks.

6 Go to frame 10, and press F9 to render the Perspective viewport.

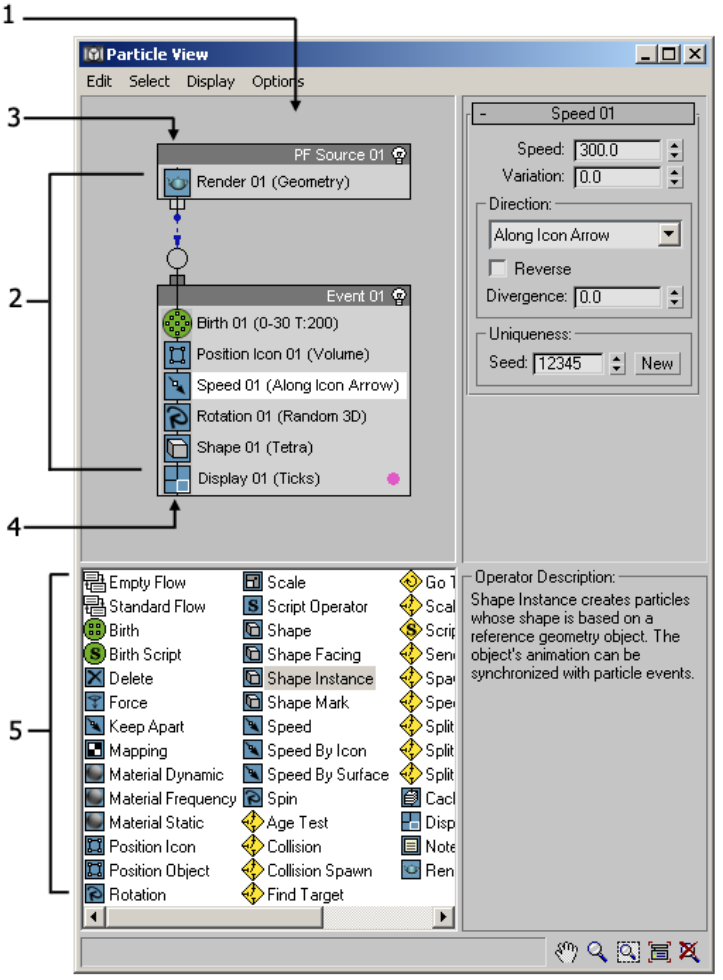


The rendered particles appear in a window. The default particle shape is a tetrahedron, a four-sided triangular solid. Its geometry is very simple, so the system can handle many particles quickly and efficiently, but it gives a good idea of how particles are behaving. Other basic shapes are a low-poly sphere and a cube; Particle Flow also lets you use any scene object as particle geometry.

Modify the particle system in Particle View:

1. Press the 6 key to open Particle View. The particle source icon need not be selected.

TIP You can also open Particle View from the command panel when a Particle Flow source icon is selected.



1. Event display

2. Particle diagram

3. Global event

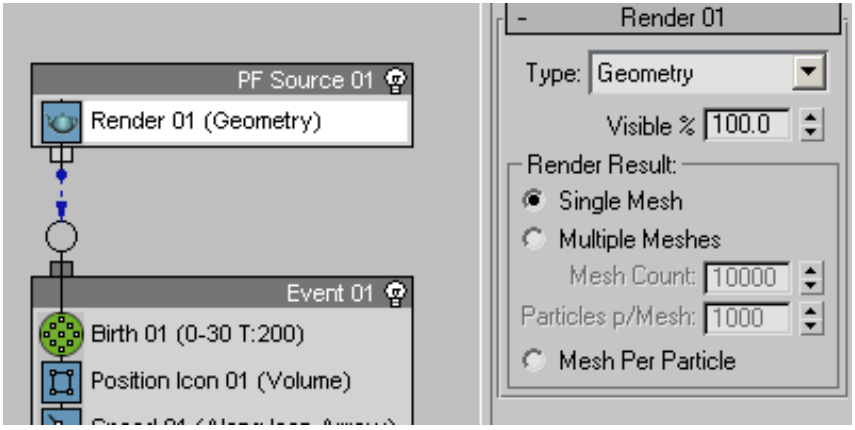
4. Birth event

5. Depot

Particle View is where you manage and edit the particle system. The main window, known as the *event display*, contains the *particle diagram*, which describes the particle system. The default system starts with a *global event*, with the same name as the Particle Flow icon, followed by a *birth event*, containing a Birth operator and several others that define the system's initial properties. Each operator's name is followed by its most important setting or settings in parentheses. Above the event display is a menu bar, and below is the *depot*, containing all operators available for use in the particle system. For more information about this dialog, see *Particle View*.

2 In the global event, PF Source 01, click the name of the Render 01 (Geometry) operator to highlight it and access its parameters. Be sure to click the text area; clicking the icon simply toggles the operator's active state.

Because this operator is in the global event, it affects the entire particle system. Any operator you place here does likewise. For example, you can define a global material here, or you can define different materials locally in each event. For best results, don't combine global and local operators of the same type in a single system; use one or the other.



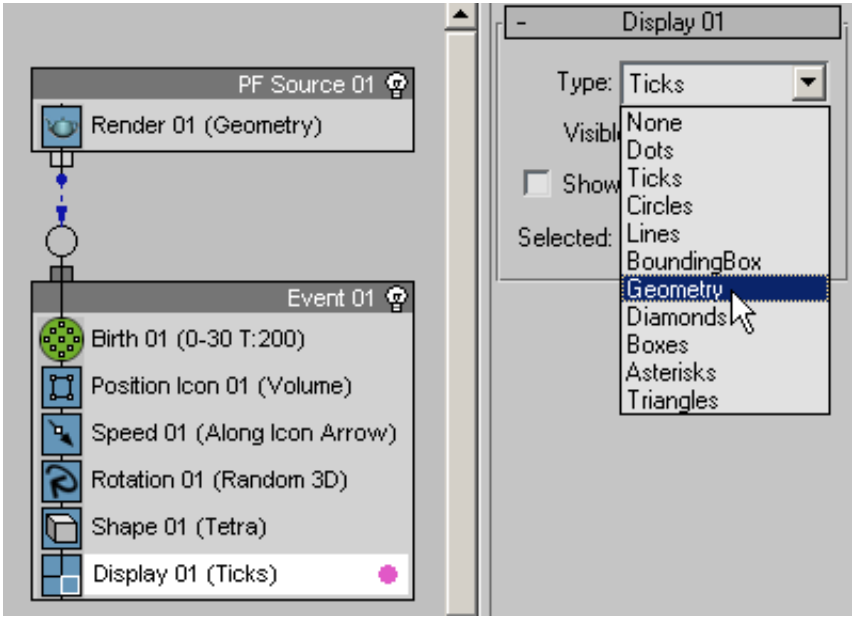
The settings for the Render operator appear in a rollout on the *parameters panel*, on the right side of the Particle View dialog. These include a drop-down list for how particles are to render, the percentage of particles that will render, and how to split up the particles into individual meshes.

Next, you'll change the particle display type.

3 In the birth event, Event 01, click the Display 01 (Ticks) operator at the bottom of the list.

The display type is set to Ticks, as noted in the operator name in the event.

4 Next to the Type label, click Ticks, and from the drop-down list, choose Geometry.

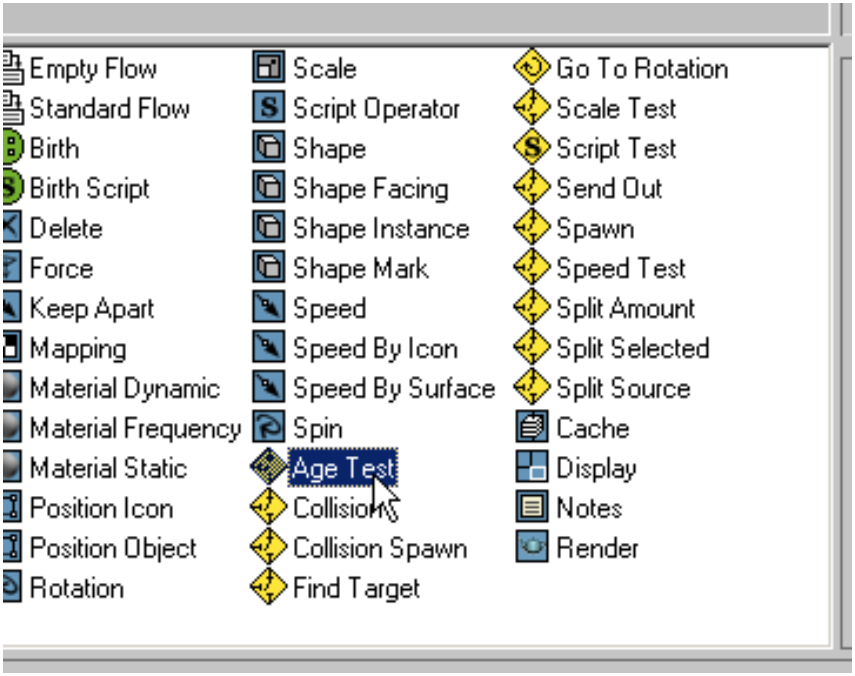


Now the particles appear as tetrahedrons in the viewports.

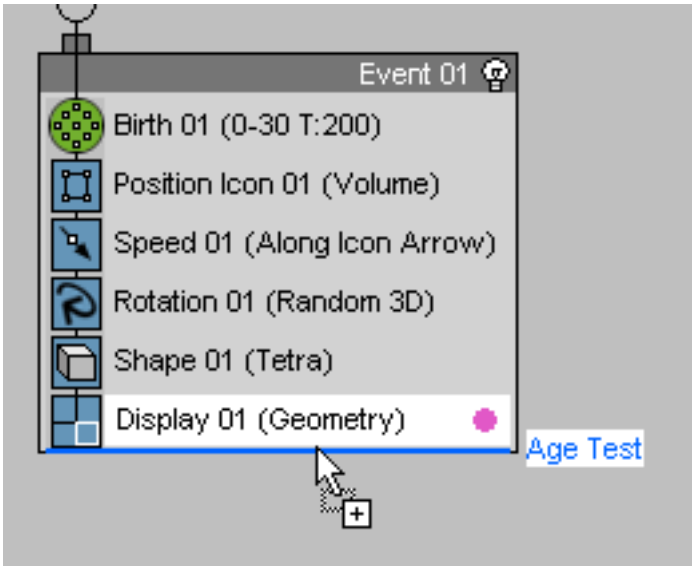
Wire two events together:

Next, you'll add a test and use it to wire the birth event to a new event.

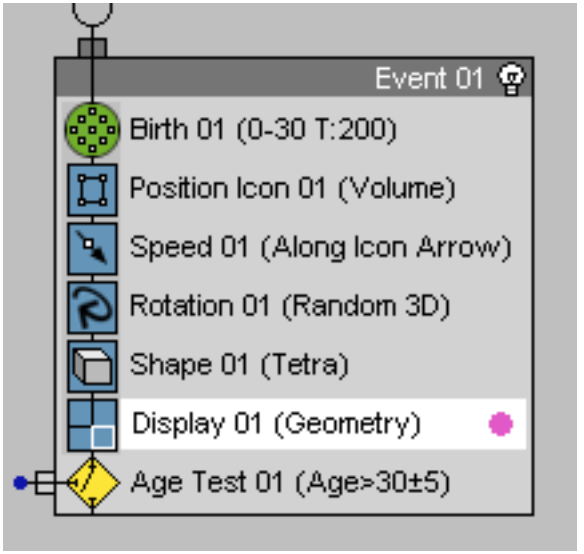
1 In the depot at the bottom of the Particle View dialog, find Age Test. It's the first item that uses a yellow, diamond-shaped icon.



2 Drag an Age Test from the depot into the Event 01 list, at the bottom of the list.

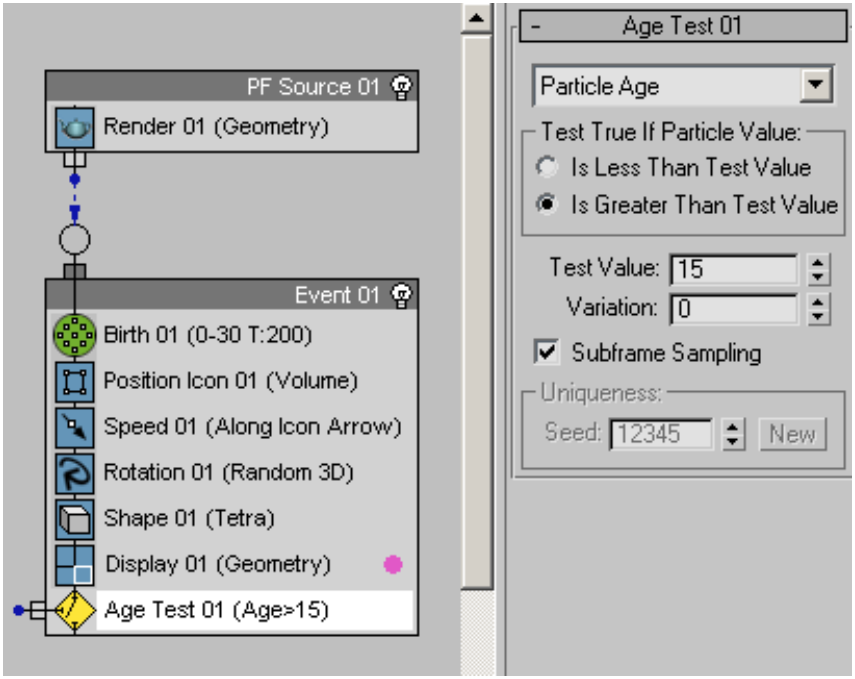


Before you release the mouse button, make sure you see a solid blue line in Event 01 under the Display operator. If the line is red and goes through an existing operator, the Age Test will replace that operator. If you drop the Age Test outside of Event 01 it will create a new event.



Age Test appears in the list, with its *test output* sticking out to the left. This is the part you use to connect the test to the next event.

3 Click the Age Test item in the list, and then in the Age Test 01 rollout on the right side of Particle View, set Test Value=**15** and Variation=**0**.



The test type is Particle Age, so this means that all particles that have existed for more than 15 frames will test True, and be passed on to the next event.

Next, you'll create a new event and wire it to the test.

4 From the depot, drag the Shape operator (“Shape”) to an empty part of the event display, below Event 01.

The Shape operator appears in a new event, named Event 02. Like Event 01, the event has a circular *event input* sticking out from the top. Also, Particle Flow automatically adds a local Display operator to the event so its particles will be visible in the viewports. You can disable the automatic creation of local Display operators by choosing Options menu > Default Display > Global.

The actual location of an event in the event display doesn't matter; the recommended placement is for the sake of convenience when wiring the events. It also helps to make sense of complex schematics if the events are arranged logically.

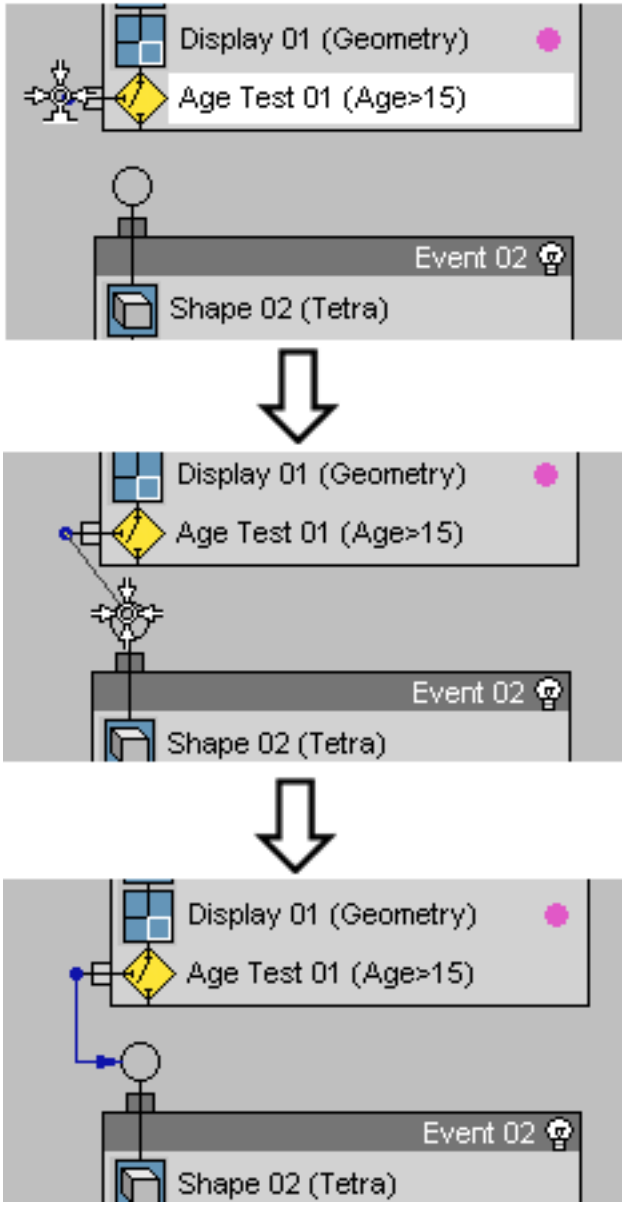
You can move an event by dragging its title bar.

Next, you'll perform the actual wiring of the Age Test to the new event.

5 Position the mouse cursor over the blue dot at the left end of the Age Test's test output.

The cursor image changes to an icon depicting three arrows pointing inward toward a circular connector.

6 Drag from the event output on the Age Test in Event 01 to the Event 02 input, and then release the mouse button.



As you drag, the cursor image changes from a three-arrow icon to a circular connector by itself when it's over an empty space in the event display area, and then to a four-arrow icon when it's over the Event 02 input.

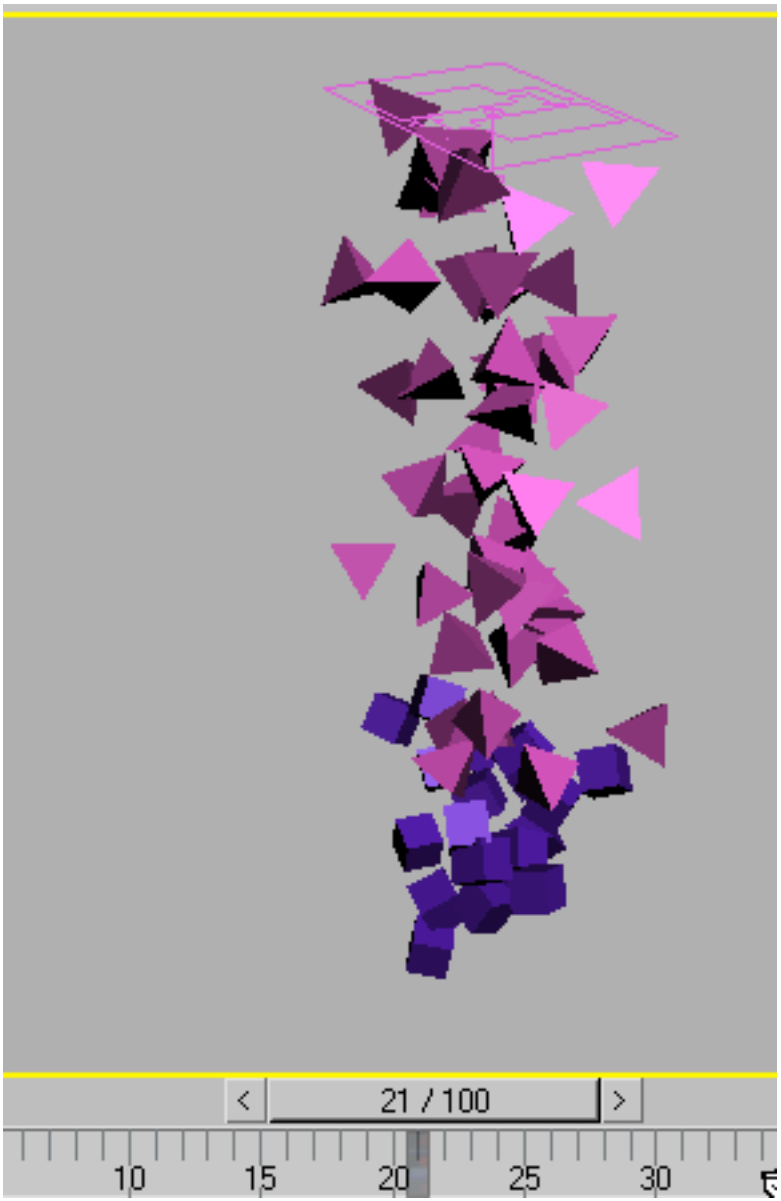
When you release the mouse button, a blue “wire” appears connecting the two events. This wire indicates that particles that meet the Age Test conditions will pass “through” this wire to Event 02, to be affected by its actions.

TIP You can delete a wire (and anything else in the particle diagram) by selecting it and pressing the Delete key. Feel free to try this now, but undo any changes when you're finished.

7 Click the Shape 02 operator and set Shape=Cube. Also, click the Display 02 operator and set Type=Geometry.

View the wiring results:

1 Play the animation. Adjust the viewports so you can see the entire particle stream, if necessary.



Starting at frame 16, the particles at the head of the stream change into cubes, indicating that they've entered Event 02. As time goes on, more

and more particles pass the age of 15 and become eligible to go to the next event.

2. While the animation is playing, try modifying the different operator settings to see what happens. For instance, click Speed 01 and change the Speed and Direction settings. When you change a setting, the change is reflected in the viewports in real time, even during playback.

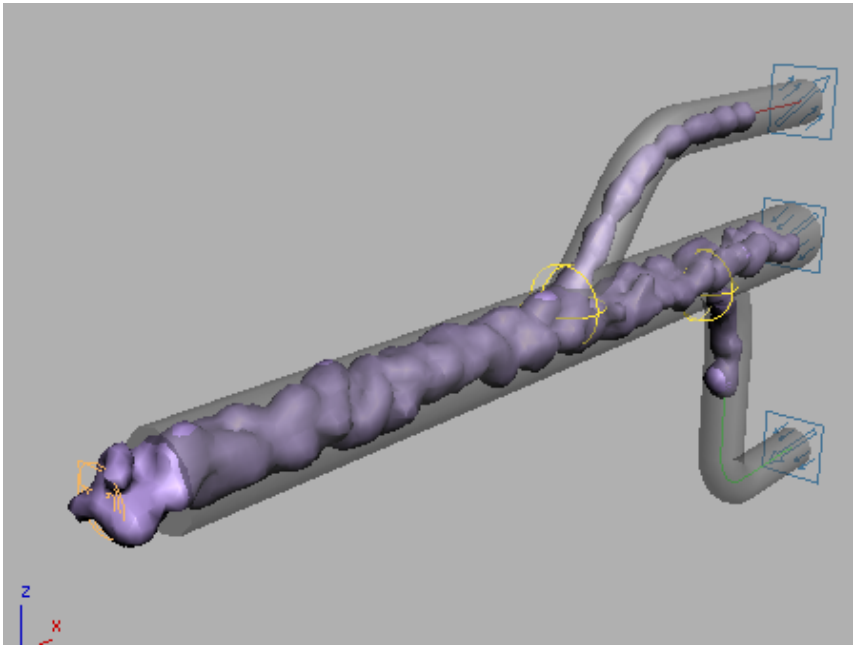
   Try right-clicking actions and events and using Rename to give them custom names. Even with a custom name, an action's tooltip reveals its type.

   Another facility of the right-click menu is to add comments to actions and events. Once you do so, a small red triangle indicates the comment's presence.

Summary

Congratulations! You now have a working knowledge of Particle Flow. If you'd like to try some more tutorials, please go to *this page* and download the 3ds Max Tutorials Help (or PDF versions) and the Scene and Support Files.

Visualizing Flow Through Tubes



In this tutorial, you'll use Particle Flow's event-driven features to control the flow of a simulated fluid substance (gas or liquid) through several branching tubes. Arrows visualizing the direction and speed of the substance will be animated as particles. As arrows approach branches in the tubes, they will selectively change flow direction, and, ultimately, will change color as well. You'll also learn how to use the BlobMesh compound object to simulate the look of the fluid substance for rendering purposes.

NOTE This is an intermediate-level tutorial. You should be familiar with basic Particle Flow and 3ds Max functionality such as track bar usage and the Align tool.

Skill level: Intermediate

Time to complete: 45 minutes

In this tutorial, you will learn how to:

- Set up an event-driven particle system
- Use instanced geometry as particles

- Create branching paths
- Use BlobMesh compound objects to create a viscous fluid.

Setting Up the Particle System

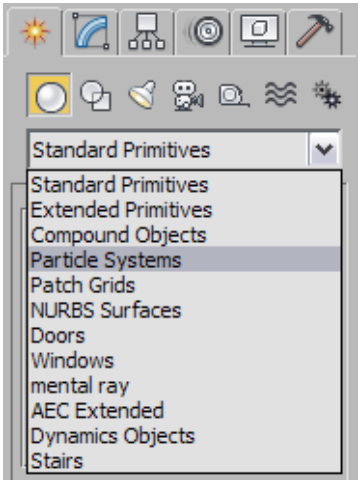
Create the gas source:

In this procedure, you'll create the source of the gas using Particle Flow.

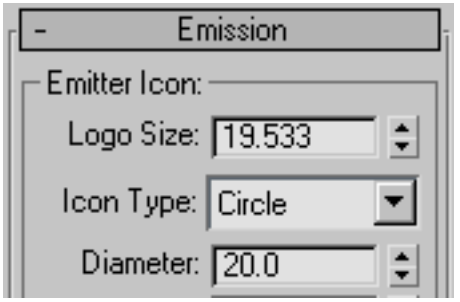
1. On the Quick Access toolbar, click the Open File button, navigate to the *\scenes\dynamics_and_effects\mech_design* folder and open the file *VizGasFlow_start.max*.

   The scene contains a branching tube object, named *TubeStructure*, set to See-Through so that you can see objects inside the geometry in shaded mode. It also contains three splines that define different paths through the tubes. You'll create a source for the particles to emit from on the left side or entrance to the tube.

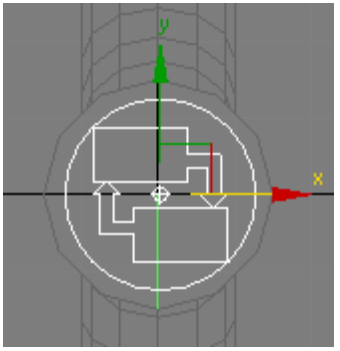
2. Go to the Create panel. From the drop-down list, choose Particle Systems, and then click PF Source.



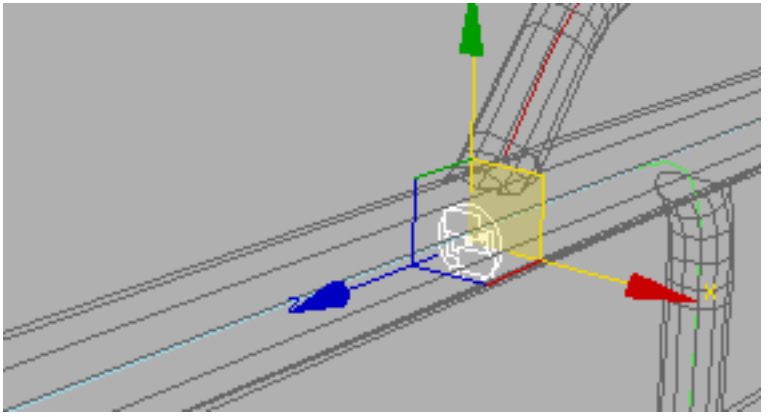
3. On the Emission rollout, set the Icon Type to Circle with a Diameter of **20**.



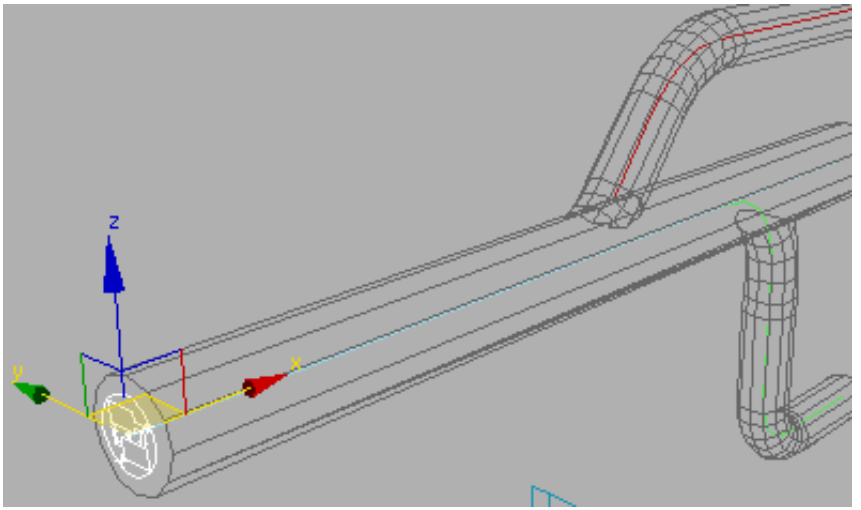
4 In the center of the Left viewport, drag out a PF Source icon of about the same size as the tube opening.



In the Perspective viewport, you can see the PF Source icon displayed as a circle with a logo and an arrow pointing outward from the center of the icon.



5 Move the PF Source icon so that it's to the left of the TubeStructure mesh, as shown below. The X axis arrow should be pointing into the tube.



Define the default flow:

In this procedure, you'll start building a flow within Particle View that will describe how particles will behave within the scene. In particular, you'll define the default flow and direction of the particles.

All particle systems created with Particle Flow use a special editor called Particle View that lets you define the behaviors of particles in a scene. When you create a PF Source icon in your scene, the software automatically creates a standard flow within Particle View. You'll see this when you first open Particle View.

1 With the PF Source icon selected, go to the Modify Panel and click the Particle View button (or press the 6 key).

This opens the Particle View editor in its default arrangement, showing the four main sections: event display, the parameters panel, the depot, and the Description panel.

The event display is where you work with operators and events: the components that define particle behavior in a Particle Flow system. The default standard flow consists of a PF Source event and a generic event that contains operators to define a standard stream of particles: Birth, Position, Speed, Rotation, Shape, and Display.

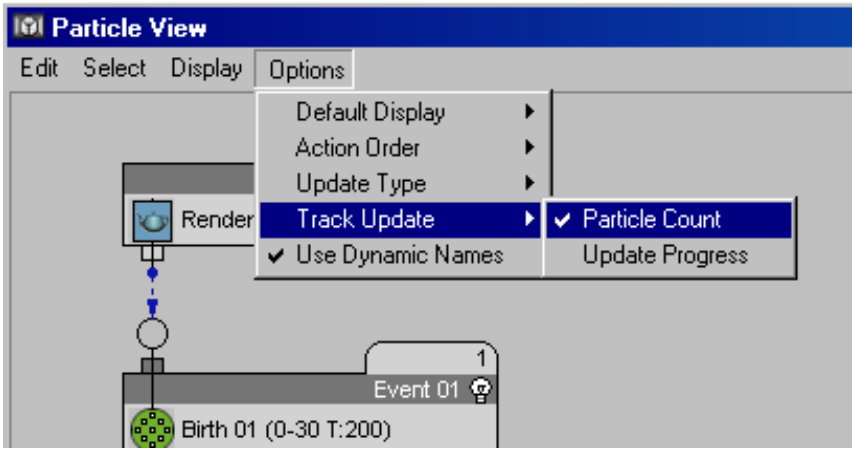
The parameters panel becomes active when you have an action (operator or test) highlighted by clicking it within an event. You can highlight multiple actions to create a stack of parameters for easy access and organization.

The depot is a storage area for all actions available to Particle Flow. You can drag actions from the depot to events; you can also add them via the right-click menu.

The Description panel goes with the depot, although it's displayed separately. When you click an action in the depot, the Description panel displays a text description of the action's function.

If you play the animation at this early stage, you'll notice that this standard flow just emits a bunch of particles in the direction of the icon's arrow from frames 0 to 30.

2 In Particle View, open the Options menu and choose Track Update > Particle Count.



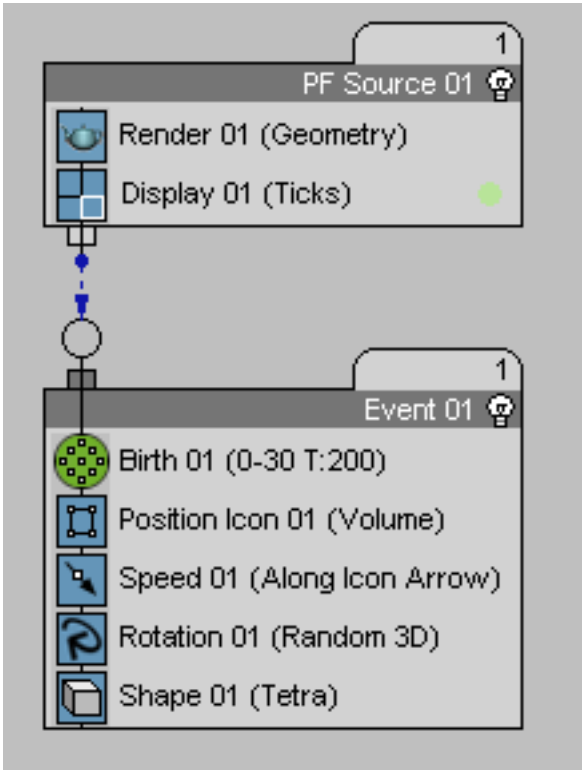
A tab with numbers appears over each event. This is a useful method for seeing how particles are moving through the events within your flow. If you play the animation or drag the time slider, the tab updates with the number of particles in each event. The global event (PF Source 01) shows the total number of particles in the flow, which at this point is the same number of particles as in Event 01.

Notice that each action in an event has some information listed in parentheses next to the action name. This is a quick reference to the action's settings. For example, the Birth operator reads: "Birth01 (0-30 T:200)." This means that it's a Birth operator set to emit a total of 200

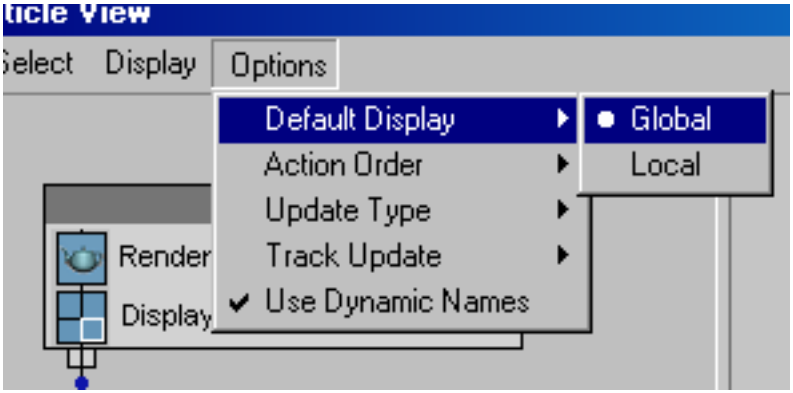
particles from frame 0 to 30. You can verify this by clicking the Birth operator and examining its settings on the Parameters panel.

Also notice that the PF Source 01 event is the initial event and contains only a Render operator. This is because it acts as a global event, defining characteristics of the entire particle flow, not individual events and operators. Any operator in this event is applied globally to all events. You can take advantage of this capability by moving the Display operator to the PF Source event so that no matter which event a particle is in, it will always use the same display method.

3 Drag the Display operator from Event 01 to the PF Source 01 event.



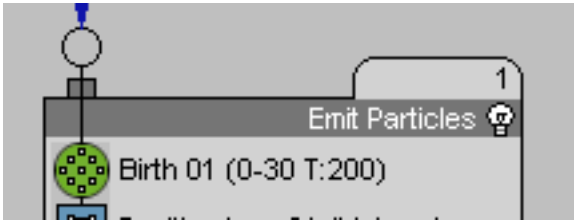
4 Choose Options menu > Default Display > Global.



This option prevents Particle View from creating new and unique Display operators for each event.

Change initial settings:

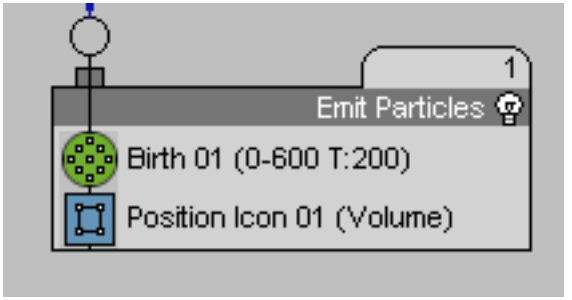
1 Right-click the Event 01 name (next to the light bulb icon), choose Rename, and enter **Emit Particles**.



2 Click this event's Birth operator, and on the Parameters panel, change the Emit Stop value to **600**.

This is the last frame in the animation.

3 In the Emit Particles event, right-click on the Speed, Rotation, and Shape operators and choose Delete.



Now the flow is much simpler. You've got a global event with Render and Display operators and a single local event with a Birth and a Position event.

If you play the animation now, you'll see that particles appear on the PF Source icon but don't go anywhere. That's because in the Emit Particle event, the position of the particles is defined by the Position Icon operator, setting particles to use the volume of the PF Source Icon.

Creating Particle Motion

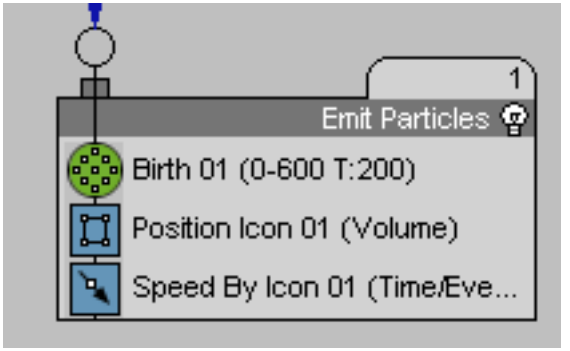
Create particle motion:

In this procedure you'll use the Speed By Icon operator and a Path constraint to define the motion of particles along a spline through the main tube.

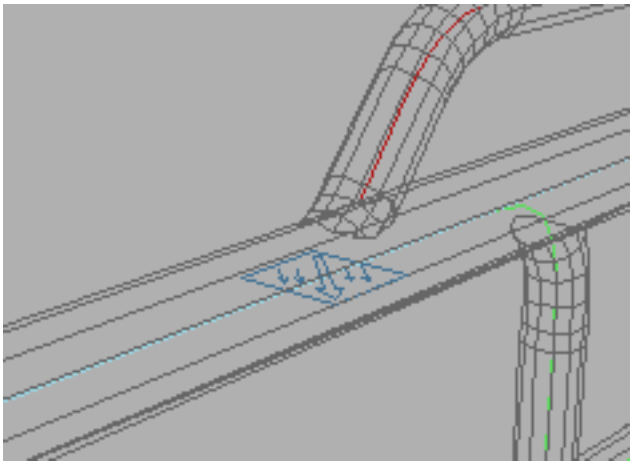
Within an event, a Speed-type operator (Speed, Speed By Icon, Speed By Surface) can give particles velocity: that is, speed and direction. Moving particles along a specific path can be difficult, but luckily there's a convenient method within Particle Flow. Using the Speed By Icon operator, you can define motion along a path that the particles can follow with a variety of controls.

1. Continue from the previous lesson or load the file *VizGasFlow_create_motion.max* from the *mech_design* folder. Open Particle View if necessary.
2. In the Emit Particles event, right-click and choose Append > Operator > Speed By Icon.

   This adds a Speed By Icon operator to the bottom of the event.



This also automatically creates a new object in your scene at the world origin (0,0,0): a helper object named *Speed By Icon 01*. If you animate this icon, the particles can inherit its motion.



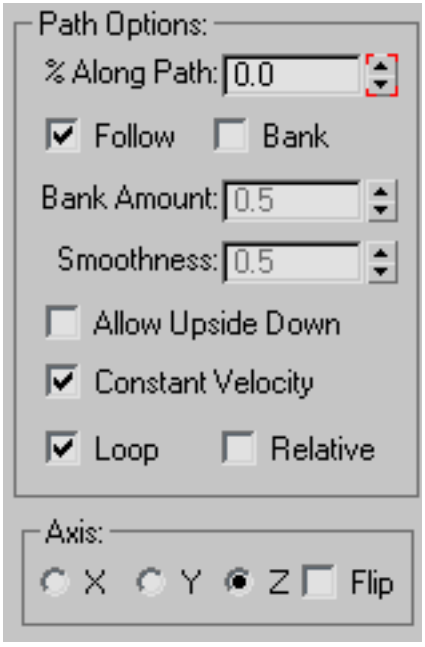
The Speed By Icon helper object appears at the world origin.

3 Press **H** to open the Select From Scene dialog, choose the Speed By Icon helper object, then from the 3ds Max main menu, choose Animation > Constraints > Path Constraint.

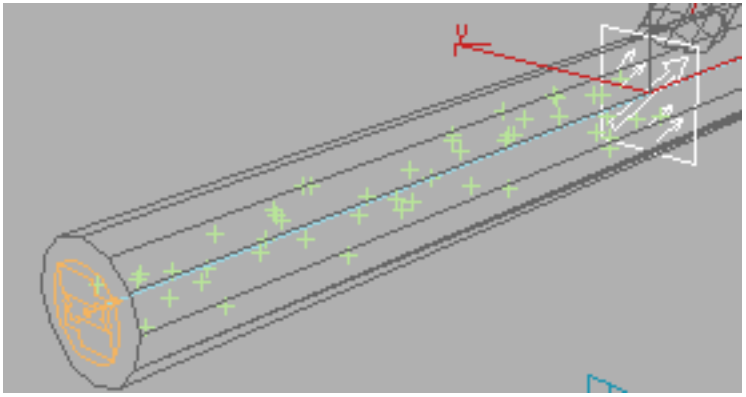
If you move the mouse in the viewport, you can see a rubber-band dashed line connecting the icon to the mouse cursor. The software is requesting a path (spline) to follow.

4 In the viewport, click the spline running through the center of the main tube (PathMain), or press **H** and then double-click PathMain.

5 On the Motion panel, which opened when you applied the Path constraint, find the Path Parameters rollout. Turn on Follow, and then, in the Axis group, choose the Z option.



The Speed By Icon 01 icon is now constrained to the PathMain spline over the course of the animation and is oriented perpendicular to the spline. If you play the animation at this point you'll see that the particles explicitly follow the Speed By Icon 01 helper object.



6 Rename the Speed By Icon 01 helper **Icon Main** for easier reference.

7 In Particle View, click the Speed By Icon operator (named Icon Main), then in the parameters panel Icon Animation group, make sure Event Duration is active.

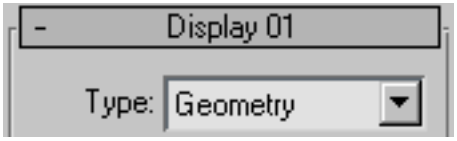
This will allow the particles to reference the animation of the icon regardless of when they were emitted, as long as they are still contained within the Emit Particles event.

Turn the particles into arrows:

In this procedure you'll use the Shape Instance and Display operators to cause the particles to appear in the scene as arrow-shaped objects.

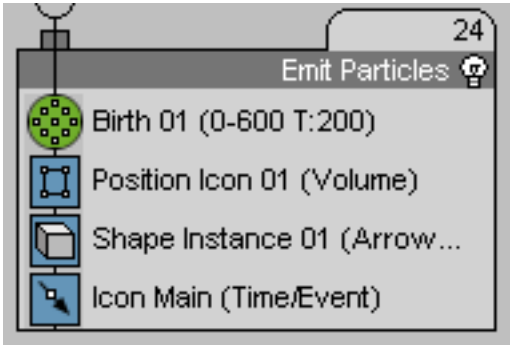
The flow currently contains Render and Display operators, but there isn't any geometry assigned to the particles. The particles appear in the viewports as arbitrary shapes, but they don't render at all. You can use the Display operator to change the way in which particles appear.

1 Click the Display operator and change Type to Geometry.



The particles now appear as X shapes. Particle Flow uses this display method to indicate an error when it is trying to display particles as geometry, but there is no geometry to display. To resolve this, you'll define the geometry for particles that are in the Emit Particles event.

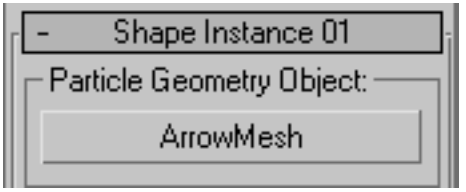
2 From the depot, drag a Shape Instance operator into the Emit Particles event. You can place it anywhere in the event, but be sure to drop it when the blue line appears; if you drop it when a red line appears, you'll replace the action under the mouse cursor.



The Shape Instance operator lets you use mesh objects in your scene as particles. With the non-event-driven particle systems such as PArray, you can only use one object at a time, but Shape Instance enables more flexibility by letting you assign groups of objects and cycle through them as particles are emitted.

3 Click the Shape Instance operator in the Emit Particles event, click the Particle Geometry Object button (currently labeled "None"), and select the ArrowMesh object in the scene.

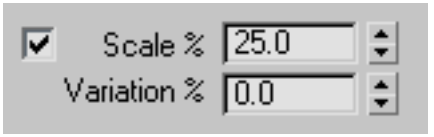
The name "ArrowMesh" appears on the button.



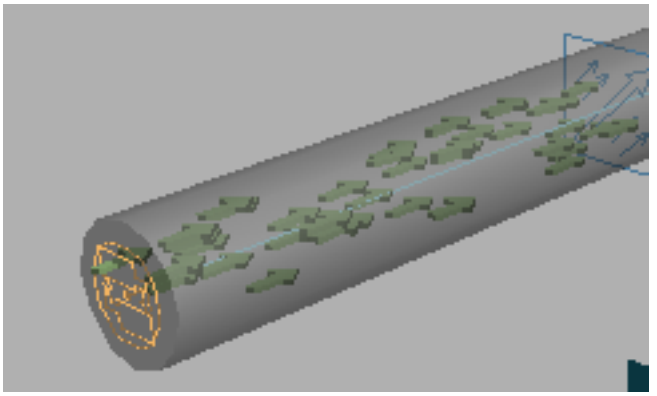
You've now defined the ArrowMesh object to be the geometry all particles will use within the Emit Particles event. Because you previously set the Display operator to Geometry, you should now see ArrowMesh particles when you play the animation. They're too big and not aligned properly, but both conditions are easy to fix.

To make the particles smaller, you could assign a Scale operator, or you could just scale the ArrowMesh object itself, but Shape Instance includes some scaling options to save you a step.

4 In the Shape Instance 01 operator parameters, make sure the Scale check box is on, then set Scale % to **25.0**.



Now the arrows are smaller and fit within the tube without crowding.

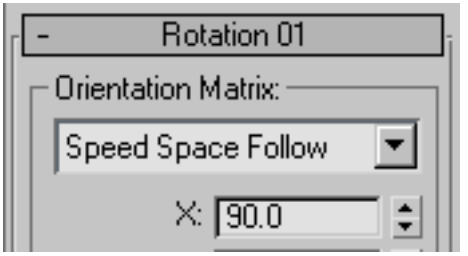


Next, you'll adjust the arrows' alignment.

5 Right-click the Emit Particles event and choose Append > Operator > Rotation from the menu.

This adds a Rotation operator to the bottom of the Emit Particles event that you can use to specify the orientation of particles in the event.

6 Click the Rotation operator, change the Orientation Matrix setting to Speed Space Follow, and then set the X value to **90.0**.



This forces the particles to constantly orient themselves in the direction they are traveling. Particle Flow provides two Speed Space methods: Speed Space sets the orientation once, when each particle first enters the event, while Speed Space Follow continually adjusts the orientation as particles move through the scene. By default, the three axis values are set to 0.0.

However, because the ArrowMesh object is pointing along the X axis, it's necessary to tell the Rotation operator to rotate particles 90 degrees about the X axis.

If you play the animation now, you should see the arrows moving correctly along the PathMain spline.

Next you'll set up tests that will cause some particles to branch off along the other splines.

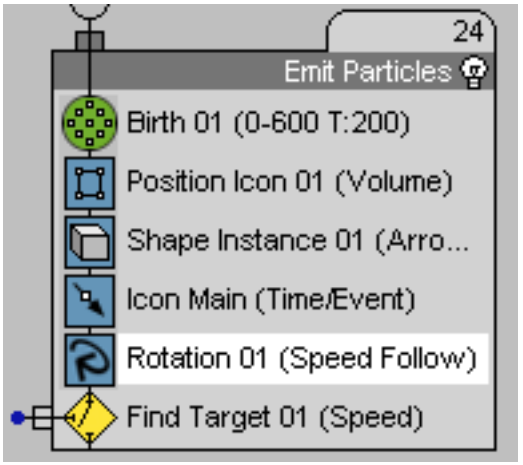
Creating Branching Paths

Create branching paths:

One of Particle Flow's unique capabilities is that it lets you create branching events that are controlled by conditions. You do this with a type of action called a *test*, which defines a true or false condition against which the software continually checks each particle. When a test returns False, the particle stays in the current event, but if the result is True the particle can move to another event that specifies different behavior.

In this procedure you'll use Find Target tests to cause some particles to change course and flow along different paths.

1. Continue from the previous lesson or load the file *VizGasFlow_create_branching.max* from the *mech_design* folder. Open Particle View if necessary.
2. Right-click the Emit Particles event and choose Append > Test > Find Target from the menu.



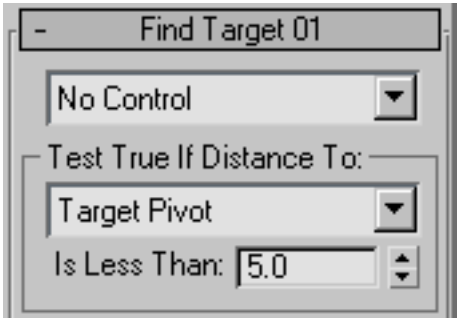
The Find Target test is used primarily to set a goal for particles. Its options include moving particles to the goal based on speed or time. In addition, you can use it as a simple proximity test, without affecting particle motion at all, as you'll do next.

3 Click the Find Target operator and, from the drop-down list at the top of the parameters rollout, choose No Control.

This disables most of the test's parameters.

When you created the Find Target test, it added a helper object to your scene called Find Target 01. This is now the target for the particles within this event; the software will test each particle based on its distance from the helper object.

4 In the Find Target test parameters, set the Test True If Distance To option to Target Pivot, and set the Is Less Than value to **5.0**.

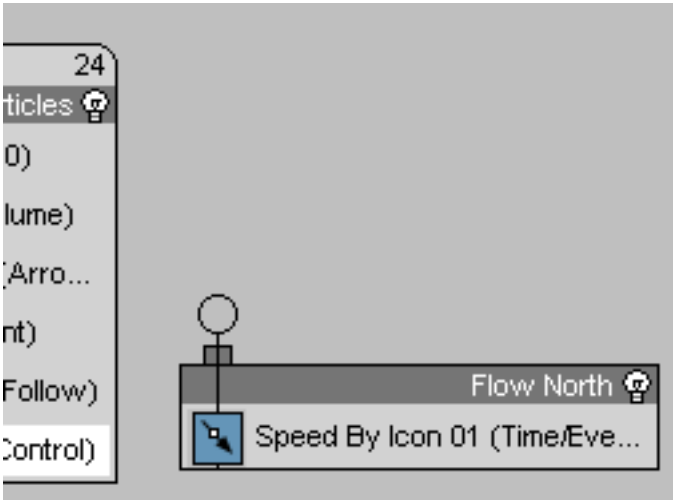


One visible difference between tests and operators is that tests have a little “nub” that sticks out to the side. You use this to wire the output of the test to the input of another event. The result is that particles that test True will go to the next event.

5 Right-click to the right side of the Emit Particles event in the blank space of Particle View and choose New > Operator Event > Speed By Icon from the menu.

You’ve created a new event containing a single operator. This is what you will connect the output of the Find Target test to. But first you'll take some simple but important organizational steps.

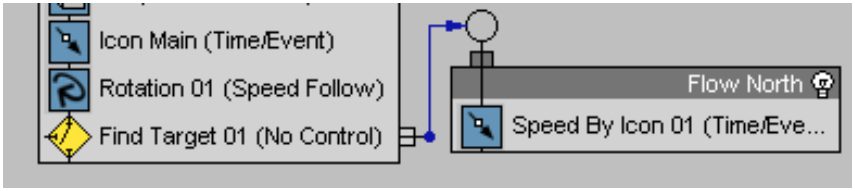
6 Rename the new event **Flow North**.



7 Move the mouse cursor to the left edge of the Find Target test, where the nub is. When the cursor changes to horizontal black arrows, click and drag the cursor to the right side of the Find Target test and release the button.

You’ve given the new event a more descriptive name, and you’ve also moved the output connector on the Find Target test operator to the opposite side so that the wire can travel in a more direct path.

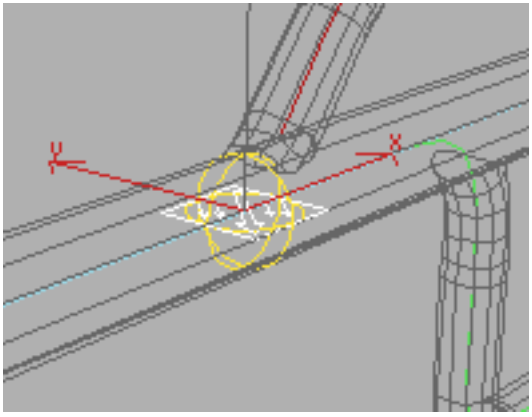
8 Click and drag from the small blue dot on the end of the test connector to the input connector of the Flow North event. The result is a blue wire connecting the Find Target test to the Flow North event.



Set up the Flow North event:

The output of the Find Target test is set up correctly, but the Flow North event isn't. The only operator within the event is a Speed By Icon operator that isn't animated. You'll animate its icon along a new path so that particles that enter this event move to that path.

1. In the parameters of the Speed By Icon operator in the Flow North event, make sure Icon Animation > Sync By is set to Event Duration.
2. Rename the operator **Icon North**.
3. Press **H** and choose the *Icon North* helper object.



Select the Icon North helper object.

4. From the Animation menu, choose Constraints > Path Constraint and complete the action by choosing the Path North spline as the path.
5. On the Motion Panel > Path Parameters rollout, turn on Path Options > Follow and set Axis to Z.

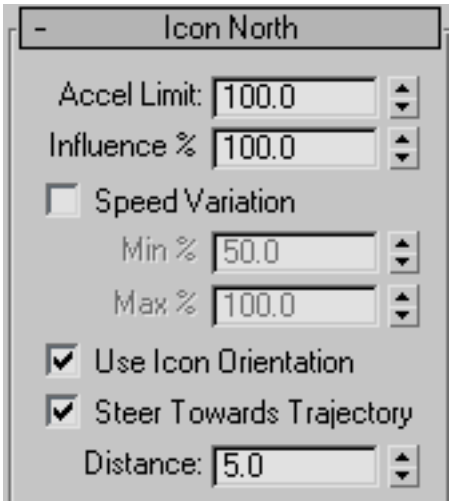
When you use a Path constraint, the software sets default keys at the first and last frames of the animation. For this animation, you'll need to adjust the keys for both the *Icon Main* and the *Icon North* helper objects.

6 In the track bar, move the rightmost key to frame 100.

7 Select the *Icon Main* helper object and slide its rightmost key to 150. This effectively increases the speed of each icon along its path.

8 In the Icon North operator parameters, turn on Use Icon Orientation.

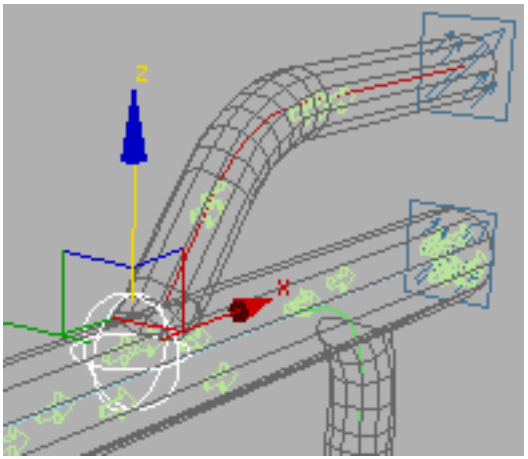
9 Turn on the Steer Towards Trajectory check box and set the Distance value to **5.0**.



TIP In addition to using Particle View, you can use the Modify panel to set parameters for a Speed By Icon operator or a Find Target test if you select its icon.

Because of the way the scene was originally set up, the Find Target icon is created close to the correct position to determine whether particles should switch to the upper tube. In the next step you'll move it upward slightly, but in many cases you'll need to reposition the target by a greater amount.

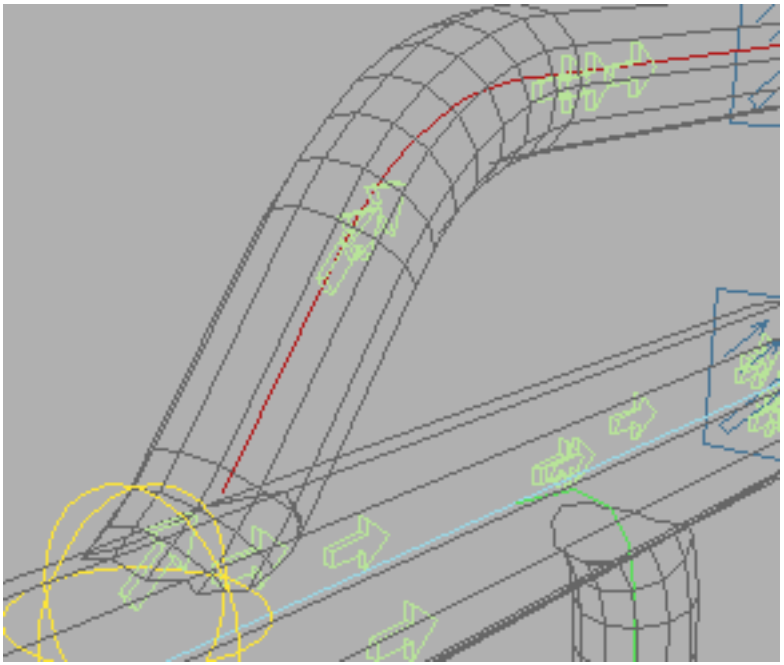
10 Select the *Find Target 01* helper object and move it 7 units upward, along the Z axis, so that it's slightly above the center line of the main path.



Some of the arrows now move upwards.

If you play the animation now, you see some of the arrows change direction and move upwards along the second path. All of the particles continue to point in the same direction because the Flow North event doesn't have a Rotation operator to define their orientation. To resolve this, you could add another Rotation operator using identical parameters, or you could instance the first one. Even better: All particles in this system should point in the direction they're traveling, so you can just move the Rotation operator from the Emit Particles event to the global event.

11 Drag the Rotation operator from the Emit Particles event into the PF Source 01 event. Be sure to drop it when you see a blue line, not a red one.



The upward-moving arrows now point in their direction of motion.

This resolves the orientation issue. We still need to address some other areas, but first you'll add one more path.

Add a third path:

Most of the following steps will probably be familiar to you.

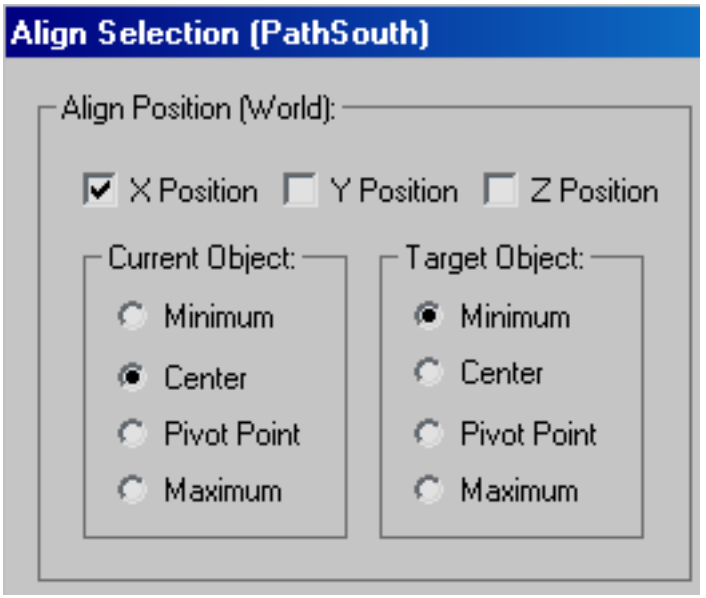
1. Append a Find Target test to the Emit Particles event.

   In order for the particles to move correctly on the third path, the Find Target icon's pivot should coincide with the path's starting point.

2. Select the *Find Target 02* icon and then press ALT+A to access the Align tool.

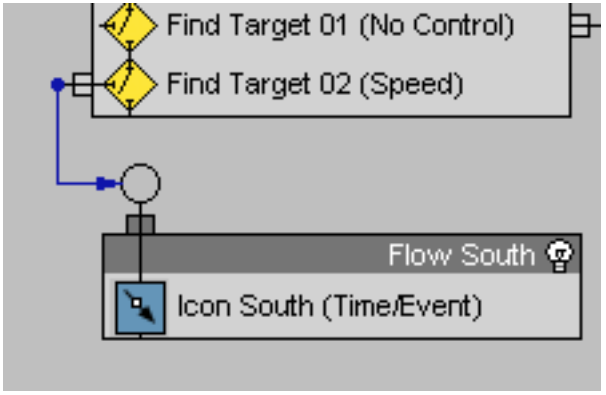
3. Select the PathSouth spline to open the Align Selection dialog.

4. On the Align Selection dialog, turn on X Position (make sure Y and Z are disabled) and set Target Object to Minimum. For Current Object, choose Center. Click OK to close the dialog.



5 Create a new event that contains a single Speed By Icon operator. Rename the event **Flow South** and the operator **Icon South**.

6 Connect the output of the second Find Target test to the Flow South event.

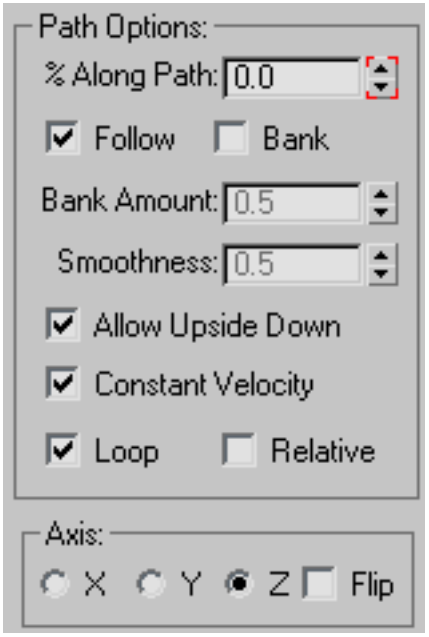


7 In Particle View click the second Find Target test, set its mode to No Control and set To Test True If Distance To: > Target Pivot to Is Less Than **5.0**.

8 In the Flow South event, set the Icon South operator to Use Icon Orientation and Steer Towards Trajectory with a Distance of **10.0**.

9 Select the *Icon South* helper object and use a Path constraint to attach it to the Path South spline.

10 On the Motion Panel, turn on Path Options > Follow and set Axis to Z. Also turn on Allow Upside Down.



This latter step helps to smooth out the motion when the particles are moving directly downward.

11 In the track bar, slide the right-hand Position key to frame 100.

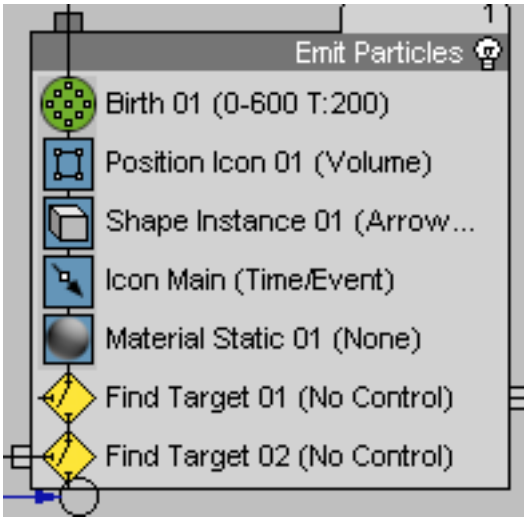
You now have icons moving down all three splines and particles following them. You might want to adjust the locations of the Find Target icons slightly so the particles branch properly.

Change colors:

What if you want to change the color of the arrows based on which path they are moving along? This is a common need in AEC visualization when showing how fluids move through pipes, with the colors differentiating the types of fluid. Because you have a different event defining each path, it's easy to assign

different materials to particles within each event. You'll use three materials that have already been prepared for you.

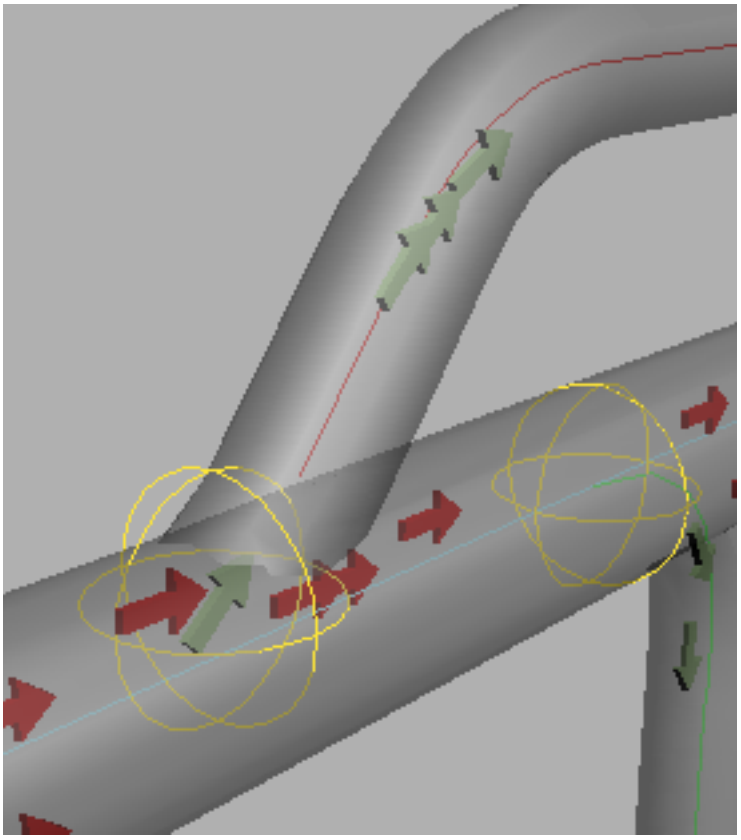
1. From the depot, drag a Material Static operator into the Emit Particles event. Place it above the tests.



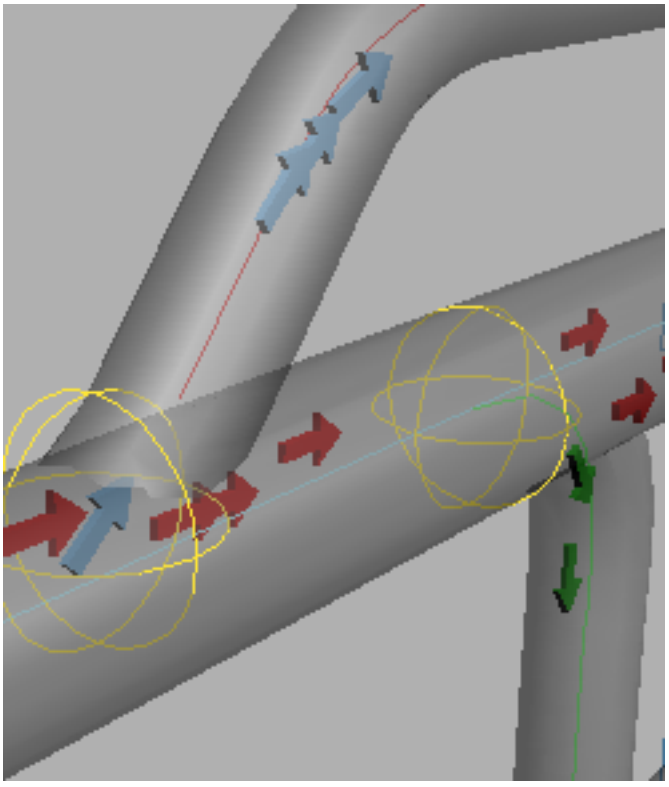
2. Click the operator to open its parameters.

3. Open the Material Editor and drag the Red material onto the button ("None") on the parameters panel. Click OK to accept the Instance method.

   If you play or advance the animation in a shaded viewport, you can see that all of the main particles are now red. The color of the particles in the other events is determined by the Display operator in the global event.



4 Add a Material Static operator to each of the two Flow events, and apply the Blue material to one and Green material to the other.



Now all three colors appear in the viewports.

Applying Final Touches

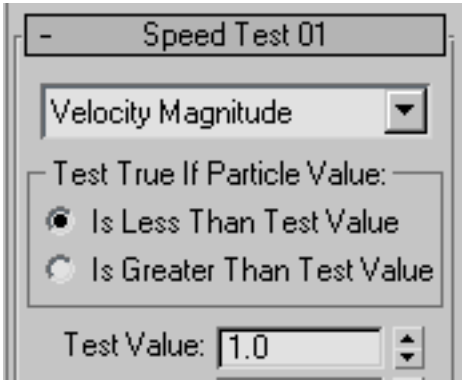
The scene is nearly complete. The particles appear as colored arrows that branch along different spline paths and change colors depending on which path they follow. One issue remains: The particles collect at the ends of the paths when they complete the Speed By Icon animation. Since this is a visualization scene and a constant flow is required, you'd ideally just want to delete particles as they arrive at the end of the paths. There are several different ways to do this, but one simple one is to just define a condition where particles are deleted when they come to a stop. While there isn't a test that's designed specifically for this, you can use the Speed Test operator and pass the result to a Delete operator. Here's how you do it:

Apply finishing touches:

1. Continue from the previous lesson or load the file *VizGasFlow_final_touches_start.max* from the *\mech_design* folder. Open Particle View, if necessary, by pressing **6** on the keyboard.

2. Right-click the Emit Particles event and choose Append > Test > Speed Test.

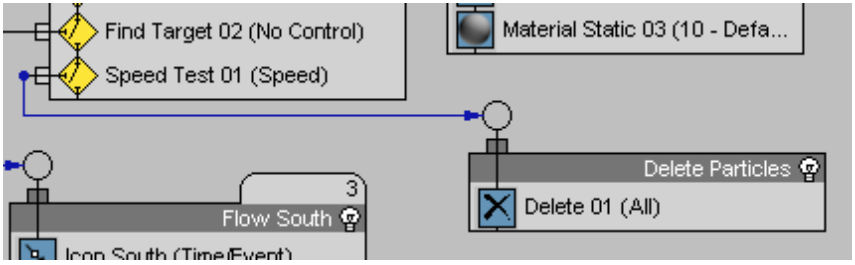
3. Highlight Speed Test 01 and set it to Test True If Particle Value > Is Less Than Test Value. Change Test Value to **1.0**.



By default, the Speed Test tests Velocity Magnitude, which is a fancy way of saying "speed in any direction." If necessary (not in this case), you can make it more specific by testing velocity along a specific axis. By setting it to "Less than 1" it will direct particles that slow to a stop to a new event.

4. From the depot, drag a Delete operator into an empty area of the event display to create a new event.

5. Rename the new event **Delete Particles**, and wire it to the output of the Speed Test in the Emit Particles event.



You've created a test for the Emit Particles event, the result of which can be seen in the animation now. The Red particles moving along the center tube disappear when they reach the end. Ideally you'd want this for the other events as well. There are several ways to accomplish this, but the easiest is just to apply the test globally.

6 Drag Speed Test 01 from the Emit Particles event to the bottom of the PF Source 01 event.

The test's wiring (to the Delete operator) moves along with it.

Now every particle disappears as soon as it comes to the end of its path.

The only potential drawback with this method is a loss of flexibility. By applying a test globally, all particles within the system, regardless of any other parameters or conditions, will be tested for speed and sent to the Delete Particles event. If you need to test for different conditions, you can include the test in each event instead.

Apply BlobMesh

Suppose your goal is not so much an abstract visualization with arrow indicators, but something more realistic. BlobMesh is a special compound object in 3ds Max that you can use to simulate fluids. It works something like the MetaParticles effect in the non-event-driven particle systems.

In this procedure, you'll use BlobMesh to simulate the flow of liquids in the upper and lower tubes.

1 On the Create panel, choose Compound Objects > BlobMesh and click in the viewport to create a BlobMesh object.

The BlobMesh object references objects in the scene, placing user-definable metaballs at each vertex in the object. When of sufficient size and density, the metaball's surface features blend together to form a continuous blob.

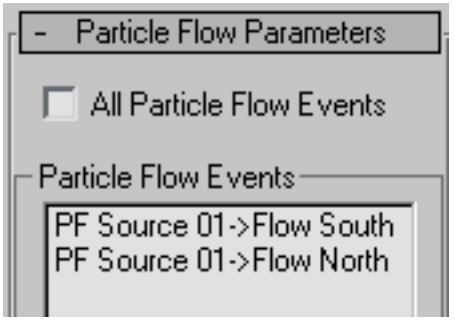
You first have to assign an object for the BlobMesh to reference: in this case, the particle system you've been working on during this tutorial.

2 With the BlobMesh object selected, go to the Modify Panel, click the Add button in the Blob Objects group and choose PF Source 01 from the list.

This applies to the system as a whole, however, so next you'll define which events to include in the blobby mesh.

3 Scroll down to the Particle Flow Parameters rollout and turn off the All Particle Flow Events option.

4 Still in the Particle Flow Parameters rollout, click the Add button and choose PFSource 01 > Flow South and PFSource 01 > Flow North events from the list. Click OK.



If you scrub the time slider now, you'll see blobs forming around the particles as they branch off into the north and south tubes. The flow doesn't look very liquid because there aren't enough particles. If you like, try increasing the Viewport % setting until you get enough particles for the blobs to flow together. You can find this setting by clicking the PF Source 01 event and then going to the parameters panel > Emission rollout > Quantity Multiplier group.

You can find the final scene, without an increased Viewport % setting, in the file *VizGasFlow_finished.max*.

An important consideration when using BlobMesh with Particle Flow is that the size of the blob is determined by the size of the particles; the BlobMesh Size parameter is ignored. You can see this by adjusting the Scale value in the Shape Instance operator. In some cases you'll need greater quantities of smaller particles. But as you increase the volume of particles, performance tends to decrease. This is because BlobMesh analyzes the particles at every frame. For faster feedback during setup of the particle system, you can turn on the BlobMesh Off In Viewport option to restrict its calculations to render time.

5 For an example of a MetaParticles scene, load and play *VizGasFlow_Blobs.max*.

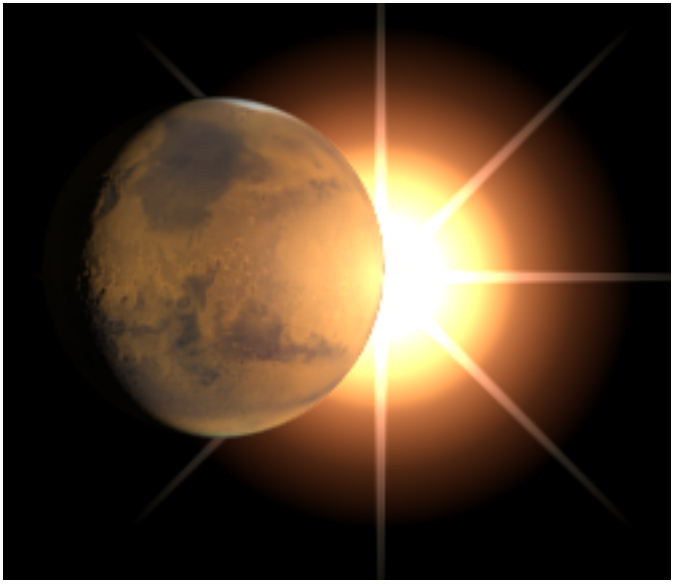
Summary

Use a particle system for simulating the flow of gaseous or liquid substances through pipes or tubing. With Particle Flow, use a Speed By Icon operator to make the particles follow a path, and use a Find Target test to divert the flow into different branches of the tubing. Use arrow-shaped particles to indicate

flow direction, or to make the flow look more realistic, apply the BlobMesh compound object.

Creating the Sun with a Lens Flare

In this tutorial, you'll create a glowing sun using Lens Flare, Glow, and Star.



In this tutorial, you will learn how to:

- Create and manipulate a camera viewport.
- Use omni lights to illiminate your scene.
- Create and manipulate a Glow lens effect.
- Use multiple Glow lens effects to create an ambient lighting effect.
- Create and manipulate a Ring lens effect.
- Create and manipulate a Star lens effect.

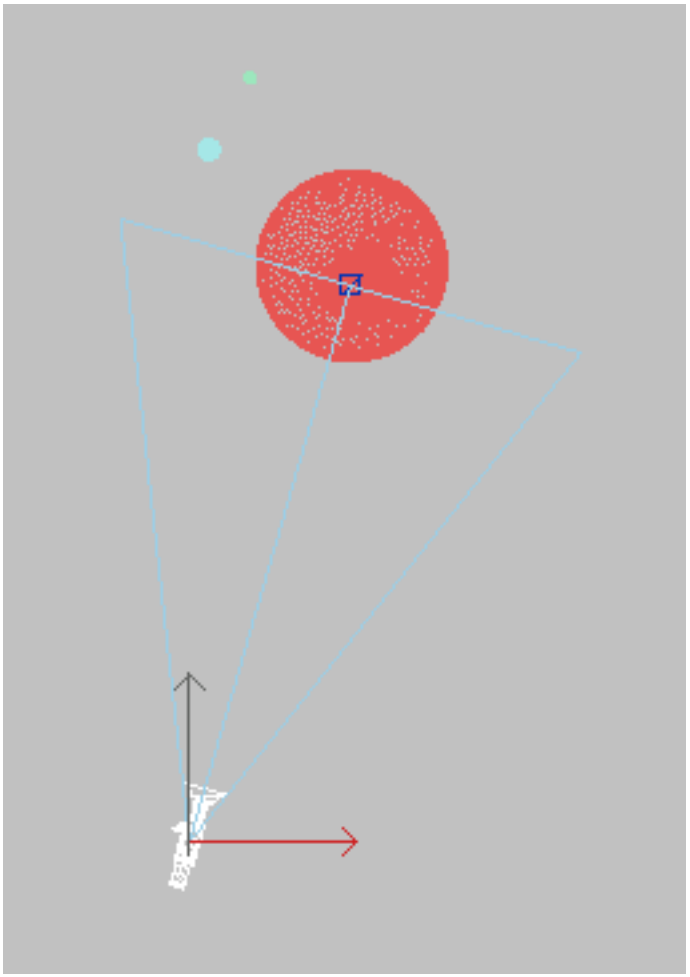
Skill level: Intermediate

Time to complete: 30 minutes

Adding a Camera View

Set up your scene:

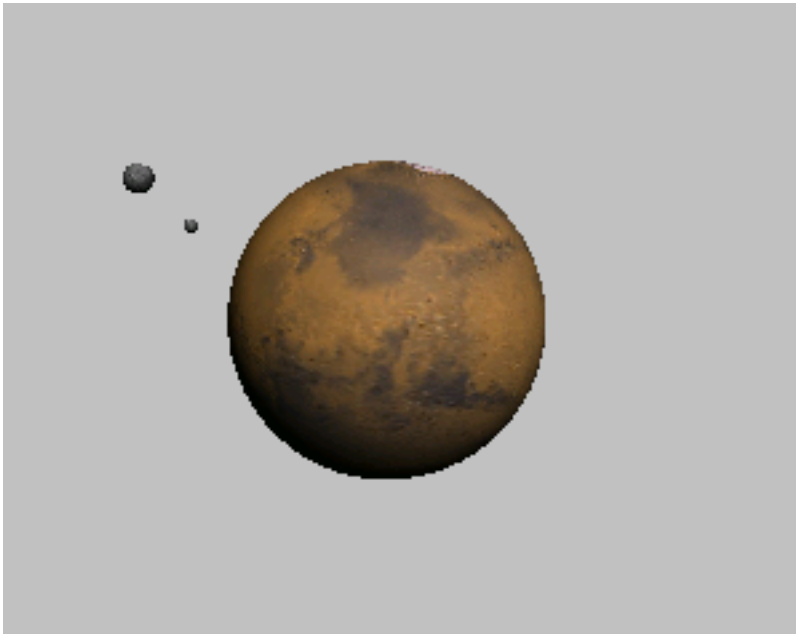
1.  On the Quick Access toolbar, click the Open File button, navigate to the *\scenes\dynamics_and_effects\lens_flare* folder, then open the file *tut_marsandsun.max*.
2. Activate the Top viewport and zoom out.
3.  On the Create panel, click the Cameras icon and click Target in the Object Type rollout.
4. Click anywhere in the bottom left of the Top viewport, drag toward Mars, and release.



This creates a target camera pointing at the center of Mars.

5 Activate the Perspective viewport. Change it to a camera viewport by pressing C.

6  Click Truck Camera. In the Camera viewport, pan the viewport to the left so there is space for the sun on the right side of Mars.

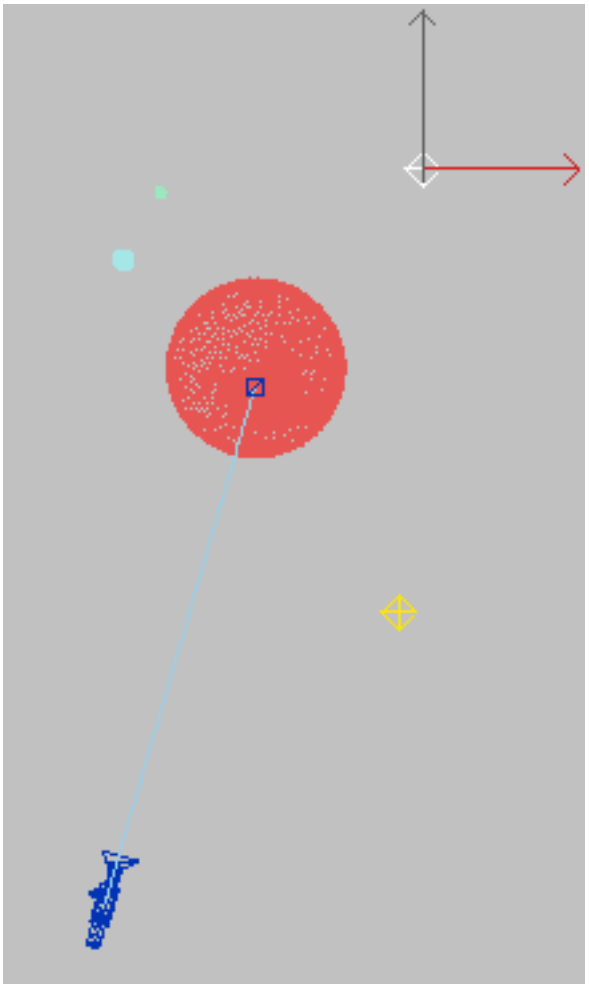


Adding Lights

There are no lights in the scene. In this step, you'll add two Omni lights: one to light Mars, and the other to be the Sun.

Light your scene:

1. From the Create menu, choose Lights.
2. From the Photometric drop-down list, choose Standard and from the Object Type rollout, click Omni.
3. In the Top viewport, click to create an Omni light to the right and below Mars. Name it **marslight**.
4. Create a second omni light to the right and above Mars. Name it **Sun**.



5 Select *Sun* and at the top of the Modify panel, click the color swatch. The Object Color selector appears. Change the color to a yellow-orange and click OK.

NOTE Both lights are adding illumination to the scene. With the camera in front of Mars, the lighting of the sun doesn't really add to the illumination on the dark side of the planet. If this becomes a problem later, you can exclude this light from illuminating any objects in the scene.

6 Move *marslight* left or right to create a lighting effect you like for the dark side of Mars.

Now use contrast to add some drama to your light.

7 In the Modify panel, go to the Advanced Effects rollout > Affect Surfaces group, and increase the Contrast setting for *marslight* to **77.0**.

You won't see the changes until you render the scene.

8  Activate the Camera viewport, then on the toolbar, click the Render Production button.

You still don't see the light that will be your Sun in the rendered image. It won't appear until you add effects in the next topic.

TIP Try different contrast values and render each one. The higher the value, the sharper the edge of the light.

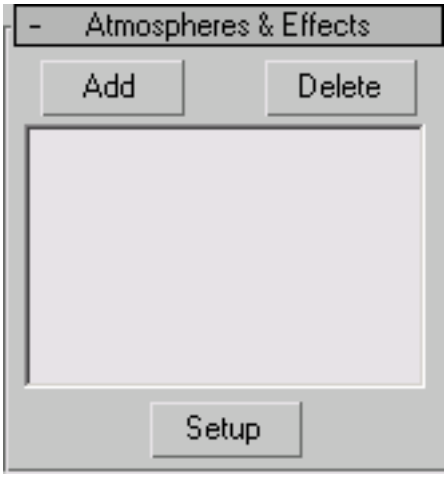
Adding a Glow to the Sun Using Lens Effects

In this topic, you'll create a simple glow effect and add a ring and a star effect to it. This will allow the Sun to glow in the sky above the planet.

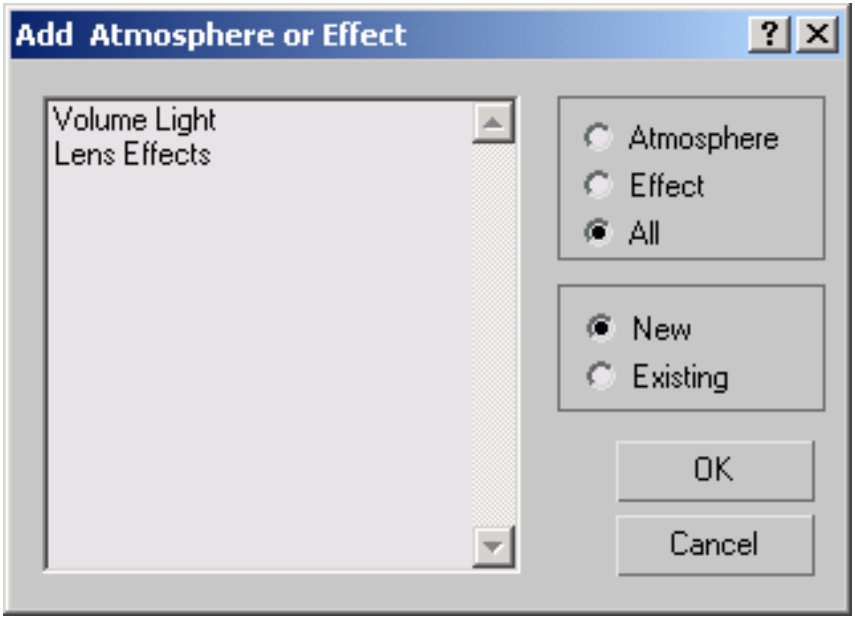
You can add as many different effects as you like to create the sun's glow. Try this effect, and then experiment with others.

Create a glow effect:

1. In the Camera viewport, select the Omni light named *Sun*.
2. Go to the Modify panel and click the Atmospheres & Effects rollout title to open it.



3. Click the Add button. The Add Atmosphere Or Effect dialog appears.



4 In the list, click Lens Effects, and then click OK.
Lens Effects is now listed in the Atmospherics and Effects window.

5 Click the Lens Effects name in the window and click Setup.
The Environment and Effects dialog appears.

6 In the Name field, name this lens effect **Sun**.

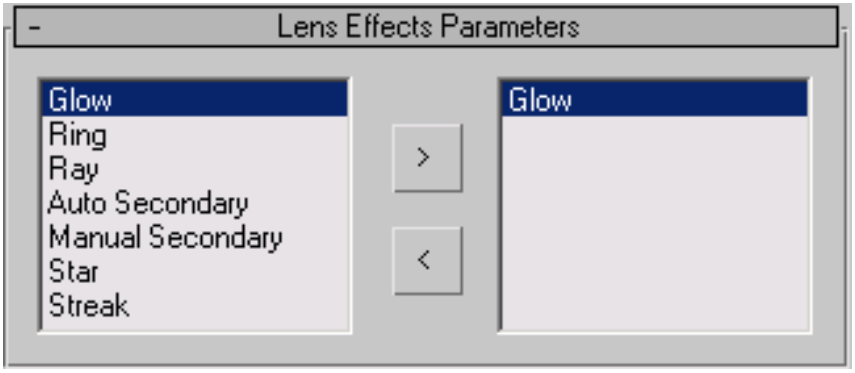
The name is changed in the windows in the rollout and the dialog.

7 In the Preview group, turn on Interactive.

A rendered frame window appears. This lets you see the lens effect as you make changes.

TIP You should turn off Interactive when working in a complex scene, but it's useful for experimentation.

8 On the Lens Effects Parameters rollout, choose Glow in the list on the left. Click the right arrow to move the effect into the list on the right.

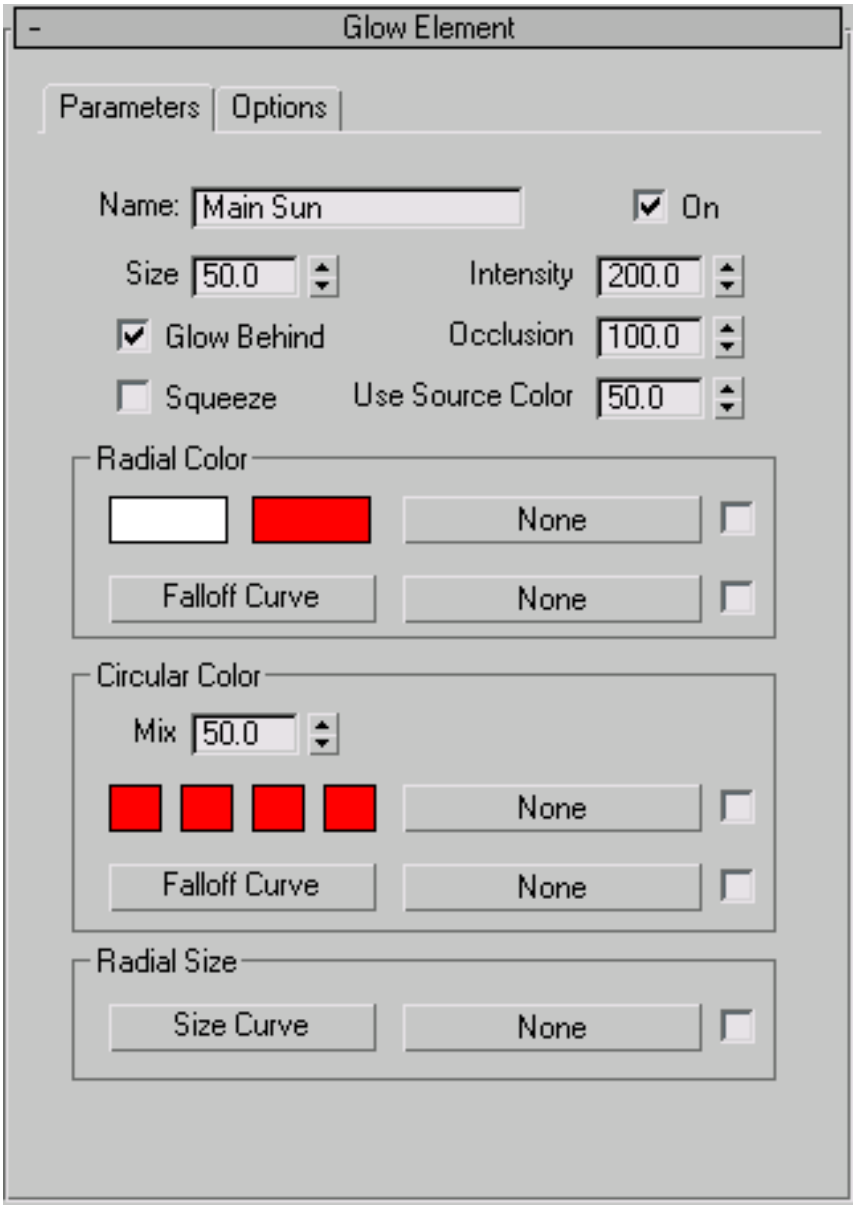


After a brief delay, the light source appears as a glowing sphere in the rendered frame.

9 Scroll down to the Glow Element rollout. In the Name field, enter **Main Sun**.

To change the look of your Sun, try the following settings in the Glow Element rollout:

- Set Size to **50.0**.
- Set Intensity to **200.0**, producing a very bright glow.
- Set Use Source Color to **50.0**.
- In the Circular Color group, set Mix to **50.0**, giving the Sun a soft red glow.


Glow Element
Parameters
Options
Name:
Main Sun
On
Size
50.0
Intensity
200.0
Glow Behind
Occlusion
100.0
Squeeze
Use Source Color
50.0
Radial Color
None
Falloff Curve
None
Circular Color
Mix
50.0
None
Falloff Curve
None
Radial Size
Size Curve
None

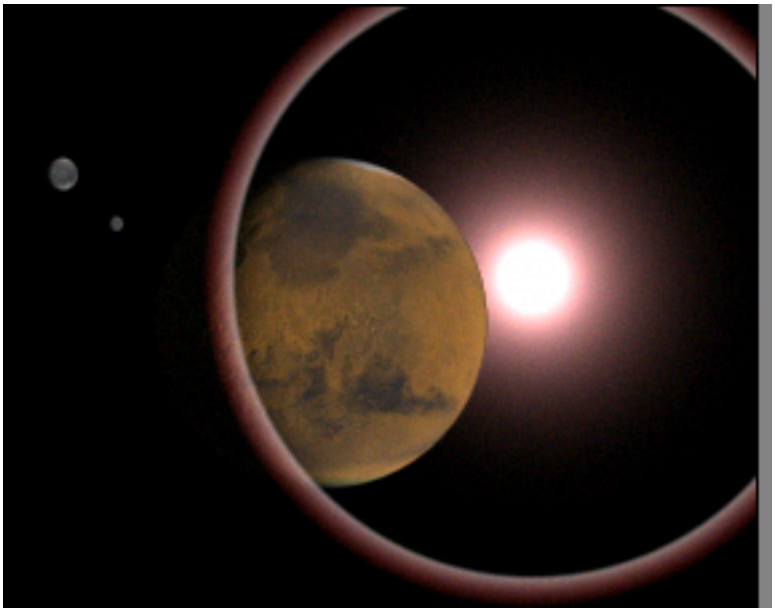
TIP With Interactive on, you'll get faster results by changing numeric settings with the keyboard, rather than using the spinners.

Adding a Ring Effect

Now you'll add a ring effect to the Sun's glow.

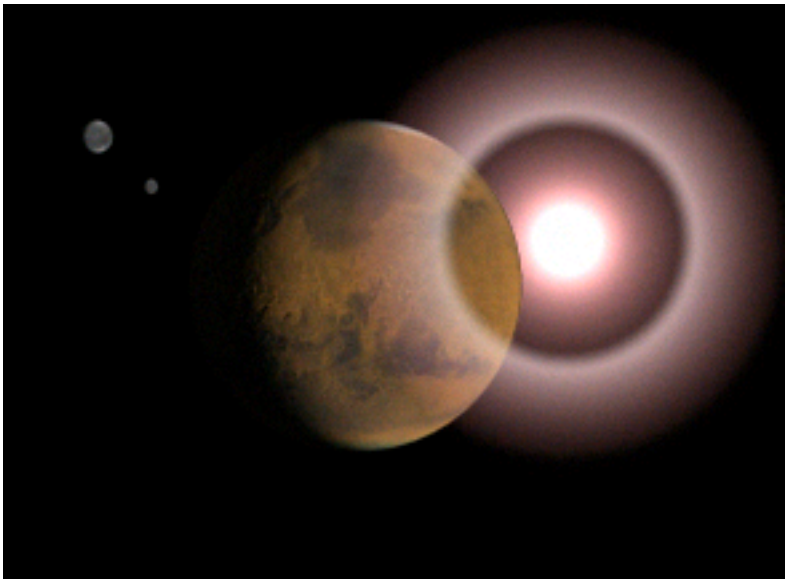
Enhance your sun with a ring effect:

1. In the Effects tab of the Environment and Effects dialog, scroll up to the Lens Effects Parameters rollout. Choose Ring and move it to the list on the right. A ring appears around the Main Sun in the Effects Preview window.



2 Scroll down to the Ring Element rollout and make the following settings to define the ring:

- Set Size to **22.0**.
- Set Thickness to **33.0**, giving the ring more of a glowing perimeter.
- Set Use Source Color to **50.0**.



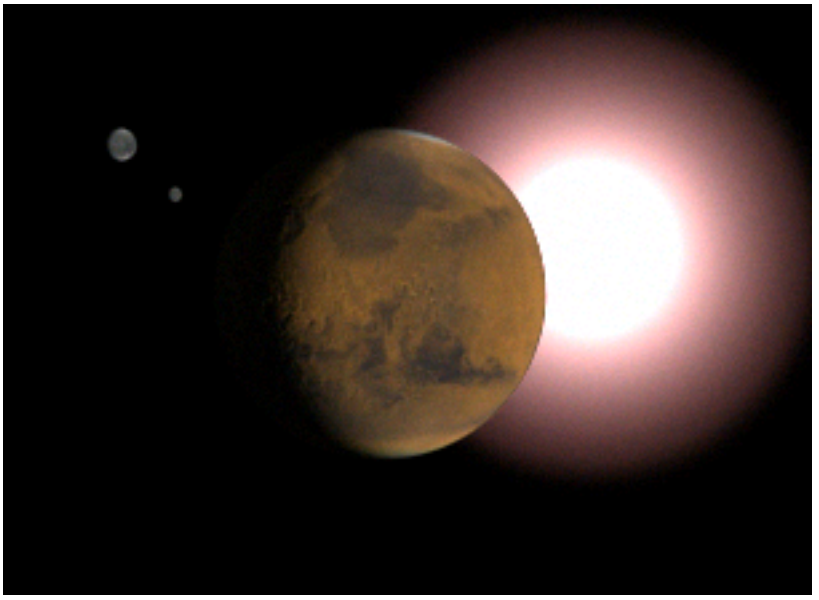
These changes make the ring more dramatic-looking, but it still needs some intensity to make it look like a glowing Sun.

Adjust the ring effect:

By increasing the intensity of the main glow and juggling the size and thickness of the ring, you can control the size of the white-hot center of the sun.

1. Increase the intensity of the Ring Element to **133.0**.
2. Lower the size of the ring to **14.0**.
3. Increase the thickness of the ring to **65.0**.
4. Turn on Glow Behind to place the glow from the Sun behind the planet.

   Now the Sun looks more realistic.

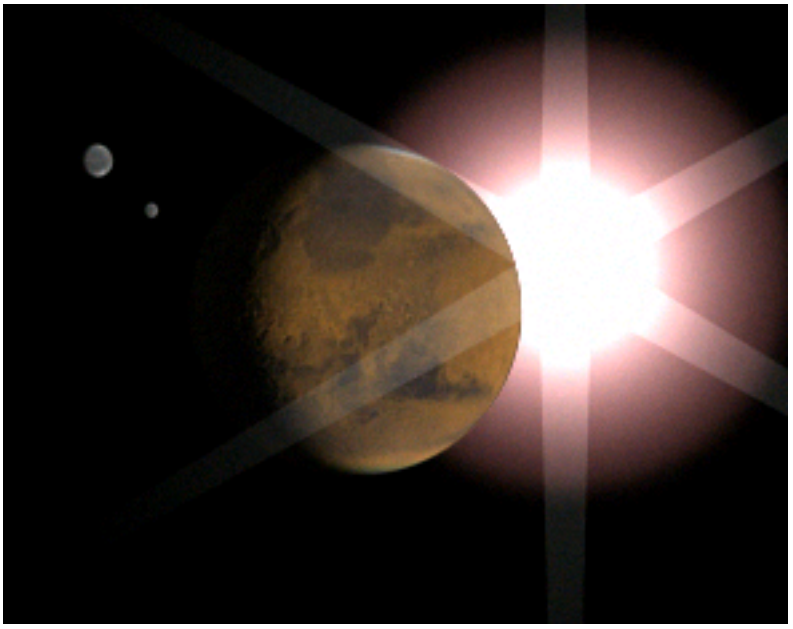


Adding a Star Effect

Now you will add a star effect to the Sun's glow.

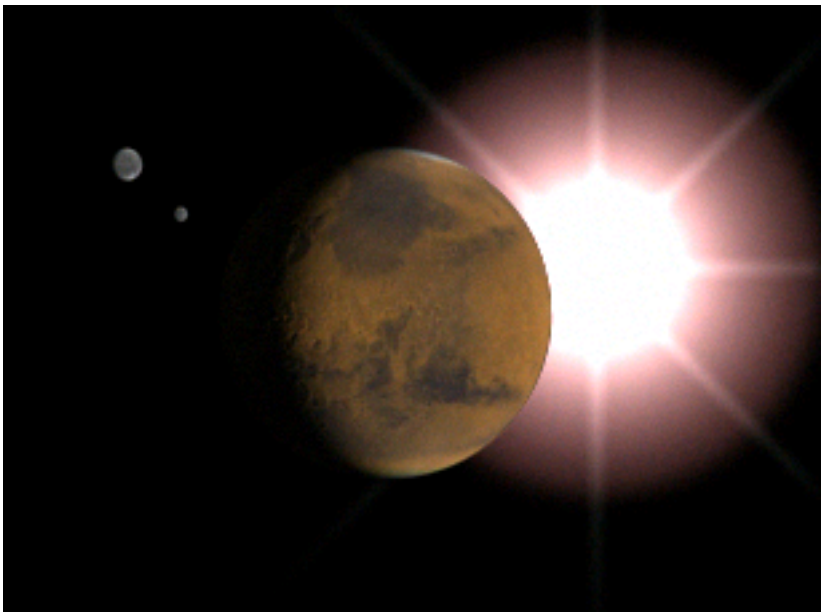
Add a star effect to your sun:

1. In the Effects tab of the Environment and Effects dialog, scroll up to the Lens Effects Parameters rollout. Choose Star from the effects list and move it to the list on the right. A Star effect appears over the Main Sun in the Effects Preview window.



2 Scroll down to the Star Element rollout, and set the following:

- Set Qty (the number of points in the star) to **8**.
- Set Intensity to **50.0**
- Set Sharp to **5.0**.
- Turn on Glow Behind.
- Experiment with Width and Taper before setting them to **1.0** and **0.1**, respectively.



You might find that your effects are incorrect if the Sun moves behind the planet. You can fix this by adjusting the effect's Occlusion settings.

Improve the star effect by adjusting occlusion settings:

1. In the Camera viewport, move the *Sun* omni light so it's just on the edge of the planet.

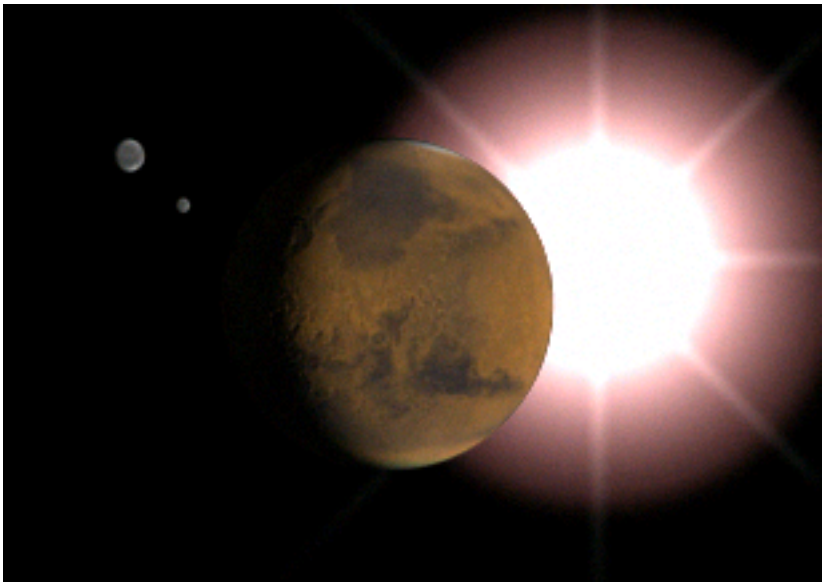
   Because Interactive is turned on, the rendered frame window updates automatically.

2. Go to the Lens Effects Parameters rollout and select *main sun* from the right-hand window.

3. Scroll down to the Glow Element rollout and set Occlusion to **0.0**.

4. Return to the Lens Effects Parameters rollout and select Ring from the effects list in the right-hand window.

5. Scroll down to the Ring Elements rollout and set Occlusion to **0.0**.

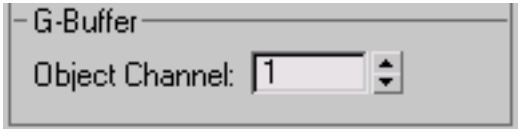


Adding Another Glow

The effects you've added to the Sun have included the Glow Behind option, so they are not adding a glow to the planet, which is in front of the light. Now that your Sun is glowing in the sky, you need to add glow to the planet itself.

Add a second glow to your scene:

1. Select the planet in a viewport. Right-click and choose Object Properties in the Transform quadrant of the quad menu.
2. In the Objects Properties dialog > G-Buffer group, change Object Channel to **1** and click OK to close the dialog.



3. In the Environment and Effects dialog > Lens Effects Parameters rollout, add another Glow to the list of effects.

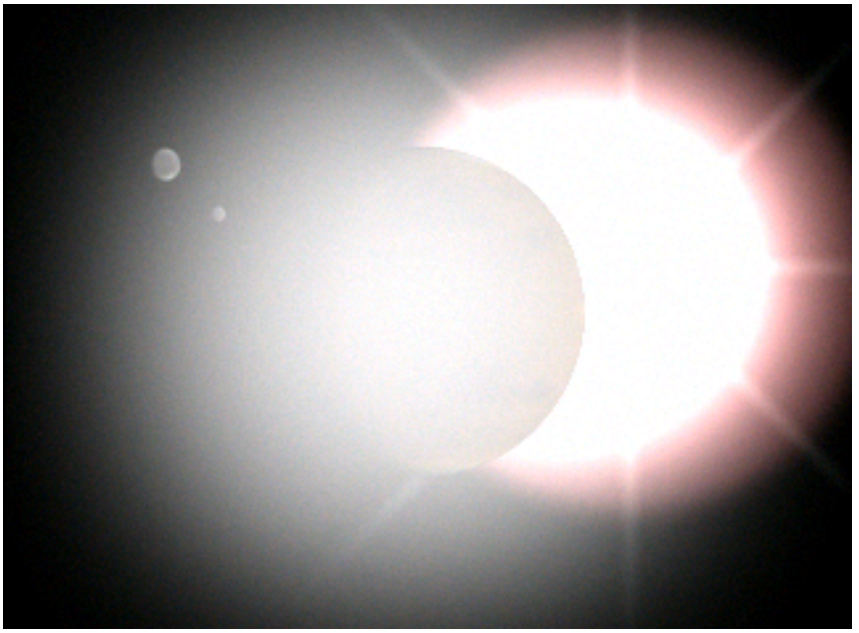
NOTE If you closed this dialog you need to select the *Sun* omni light and click Setup on the Atmospheres & Effects rollout.

4 In the Glow Element rollout, rename this effect **Glow on Planet.**

5 Click the Options tab and in the Image Sources group, turn on Object ID.

Notice that the number is set to 1 by default.

The planet now renders with a bright white glow. This is too intense.

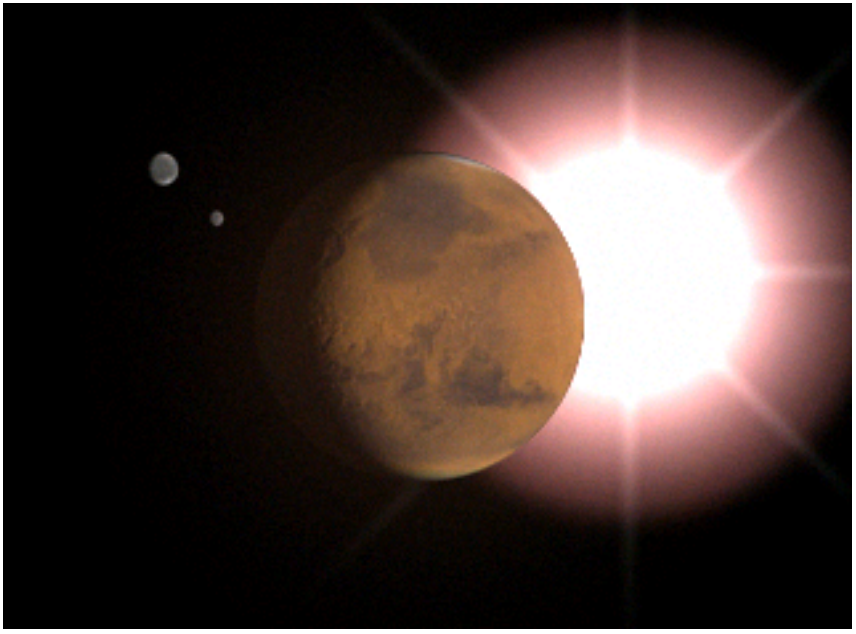


6 Click the Parameters tab. Set Occlusion to **0.0** and turn off Glow Behind.

7 Change Intensity to **45.0** and Source Color to **50.0**.

8 In the Radial Color section, change the white color swatch to a darker brick red.

The planet is looking better now.



TIP If the planet is still shadowy, move the *marslight* closer to the planet in the Top viewport.

Make global adjustments:

On the Lens Effect Globals rollout, you can make global adjustments to control the whole effect.

1. Experiment to see what happens when you change Size to **11.0**, **22.0**, and **33.0**.

   As an additional exercise, try animating the effects' settings.

2. Save the scene as **mymarsandsun.max**.

Summary

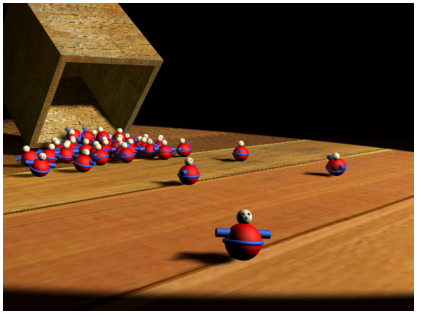
This tutorial has shown how to create Glow, Ring, and Star lens effects, and how to adjust their settings. It also showed how multiple Glow effects can create the effect of ambient lighting, how to create and navigate a camera viewport, and the use of omni lights for scene illumination.

Introduction to reactor

This tutorial will lead you through the process of creating a physically based animation using the reactor dynamics functionality built into 3ds Max.

There are many situations in animation where hand animating and creating keyframes for objects can be completely replaced; or at the very least, augmented, by using keyframes created automatically using physical techniques. For instance, imagine trying to hand-animate a thousand balls being poured out of a bucket; you would need to make sure that no two objects passed through each other during the simulation, and that they behaved realistically when colliding with each other and the environment. With reactor, however, you can produce the same animation by just creating a bucket and a thousand spheres, providing each object with physical properties such as mass. Once you have set up the objects' positions, the rest is calculated for you automatically.

In the lessons that follow, you will use the reactor functionality to create a toy with a low center of gravity. You will then create copies of the toy, and use the Preview Window to add them to a toy box and update the objects' positions in 3ds Max. Next, you will hand-animate the box turning over. The dynamics functionality will solve the animation for the toys, taking into account the hand-animated box; the toys will react to the box's motion and pour out onto the floor.



In this tutorial, you will learn how to:

- Create rigid body collections.
- Create simulations with reactor.
- Control simulation accuracy.
- Use hand animated objects with reactor.

- Create fractures with reactor.

Skill level: Intermediate

Time to complete: 90 minutes

Rigid Bodies Make the World Go Round

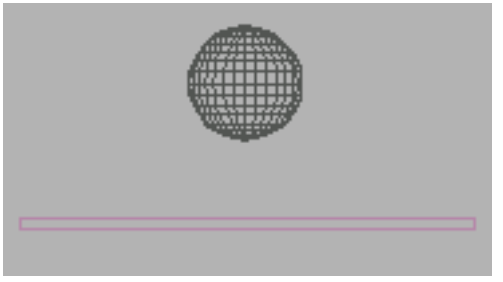
The objects in a reactor physical simulation; in this example, the toys, the toy box, and the floor, are known as *rigid bodies*. These are the building blocks of physical scenes, and can be made up of one or more geometries. You can use any geometry to create a rigid body.

In this lesson, you will create two simple primitive objects that can be used as rigid bodies, and assign them some physical properties.

Create a rigid body:

1.  On the Quick Access toolbar, click the New Scene button.
2. On the Create panel, click Sphere.
3. In the Perspective viewport, drag to create a sphere. On the Create panel, set the radius to **25.0**.
4. On the Create panel, click Box.
5. Drag in a viewport to create a box beneath the sphere. On the Create panel, set the Length and Width values to **200.0** and set Height to **5.0**.

NOTE Move the box downward to make sure it is not touching the sphere.



Now you need to give your sphere some physical properties.

6  Select the sphere, and on the reactor toolbar, click Open Property Editor .

If the reactor toolbar isn't visible, right-click an empty area of the main toolbar, and choose reactor from the context menu.

You haven't assigned any physical properties to the sphere yet, so the dialog displays default values.

NOTE If you don't have an object selected in the scene, or if you select an item that cannot be used as a rigid body (such as a rigid body collection), the dialog controls are unavailable.

7 Look at the editor's Physical Properties rollout.

The default value for Mass is **0.0**, which means that the rigid body is fixed in space during the simulation. However, in this example, the sphere is going to drop onto the box, which will act as the floor in your simulation. For this reason, you need to give the sphere a mass value.

8 Set the Mass to **50.0**.

If you do not specify any physical properties for an object, it automatically uses the default rigid-body properties when simulated. You don't want the box to move in this example, so the default properties are sufficient for this object. It uses the default Mass value of **0.0**, and is therefore fixed in space.

You now have two objects with physical properties. However, your scene is not yet valid for simulation. You must add the objects explicitly to the simulation using a Rigid Body Collection.

Getting Collective

If you have a valid rigid body and want to use it in a simulation, you need to add it to a *rigid body collection*. At the start of the simulation, reactor examines the scene for all enabled rigid body collections. It then takes the rigid bodies from the collections and adds them to the simulation. In this example, you will create a collection and add the sphere and box to it in order to add them to the simulation.

Continue using your scene from the previous lesson, or open *reactor_intro_1.max* from *\dynamics_and_effects\reactor\introduction*.

Create and add objects to a rigid body collection:

1. Make sure no objects are selected, and then, on the reactor toolbar, click Create Rigid Body Collection .

   A Rigid Body (RB) Collection is a helper object that reactor uses to keep track of the rigid bodies in your scene. It doesn't show up in renderings, and its position has no effect on your scene.

2.  Click any viewport to add the collection to your scene.

   NOTE The icon's position has no effect on the simulation.

3. On the Modify panel, in the RB Collection Properties rollout, click Pick, and then select the box in a viewport.

   This adds the box to the rigid body collection.

   You can also add objects to the collection with the Add button, as in the following step.

4. Click Add. This opens a standard selection dialog that contains a list of the remaining available rigid bodies in the scene.

   TIP To remove a rigid body from the collection, highlight it in the Rigid Bodies list and then click Delete.

5. From the list, choose the sphere, and then click Select to close the dialog and add the sphere to the collection.

TIP A useful shortcut for creating an RB Collection is to select the objects *before* you create the collection. If you select several objects and click Create Rigid Body Collection, an RB Collection is created which already contains the selected objects.

The Basics of Simulation

Now that you have a valid scene for simulation, you can try simulating it. reactor provides two methods for simulating animation:

- The reactor preview window displays your simulation using OpenGL or DirectX. You can examine your objects' physical behavior in the window, see how they interact, and even use the mouse to interact with the scene. You can use this window to update the rigid bodies' states in 3ds Max at any time during simulation, which means that it acts as an interactive scene modeler.
- Alternatively, use keyframe creation. You define an animation range, and reactor creates and simulates the physical world across that range, passing the rigid bodies' states back to 3ds Max as keyframes.

Continue using your scene from the previous lesson, or open *reactor_intro_2.max* from *\dynamics_and_effects\reactor\introduction*.

Examine your scene in the preview window:

1.  Go to the Utilities panel, click the reactor button, and, on the About rollout, make sure Choose Solver is set to Havok 1.
2.  On the reactor toolbar, click Preview Animation.

   This opens your scene in a preview window. By default, your scene is initially displayed from the Perspective view. You can use the left mouse button to rotate the camera, the middle button to pan, and the scroll wheel to zoom.
3. To start your simulation, open the Simulation menu in the preview window and choose Play/Pause, or press P on your keyboard. Your simulation starts and the sphere falls onto the box.

You can use your right mouse button to drag the sphere around the scene. However, you cannot move the box, because it has no mass.

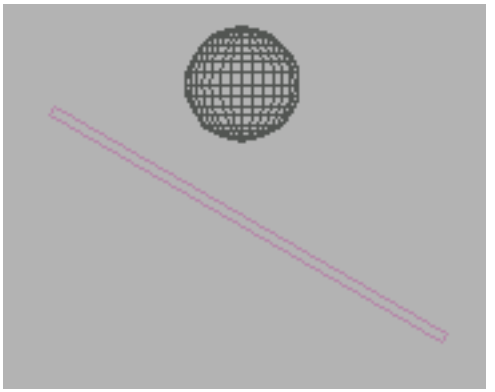
TIP To reset the simulation, press R.

4 When you're finished exploring your simulation, return to 3ds Max by closing the preview window.

Create physically-based keyframes:

Next, you will create keyframes for the objects' interaction. A sphere falling onto a level surface isn't very interesting, so first you will tilt the box so the sphere will roll along its surface.

1  In the Left viewport, select the box and rotate it clockwise about 30 degrees, being careful not to let it touch the sphere.



2  On the reactor toolbar, click Create Animation. A Reactor dialog appears, asking you to confirm the choice. Click OK to continue.

This creates a simulation and runs it for the length of time between the Start Frame and End Frame values set on the track bar. The default values for these are 0 and 100, respectively; you will change these values later in the tutorial.

reactor creates keyframes for the positions and orientations of your rigid bodies for each of those 100 frames.

3  Click Play Animation to watch the animation in the viewport. You should see your sphere falling onto the box and rolling down over the edge of it.

4 At frame 0, select the sphere if necessary, and then Alt+right-click the sphere and, from the quad menu > Set quad, choose Delete Selected Animation.

The sphere's keys on the track bar go away.

Geometry Types

Simulation Geometry is the second rollout in the Rigid Body Properties dialog; it deals with how objects are represented in the physical simulation.

In reactor, a convex object is one that has no holes or concavities in its surface. For instance, a ping-pong ball is convex, but a golf ball is concave. Convex objects are much easier to simulate than concave objects. For this reason, when you simulate an object, it is treated as if it was convex by default, even if the 3ds Max object is concave. To do this, reactor uses a special algorithm to create a convex version of the object for simulation; however, the object does not change in 3ds Max. This is called creating a *convex hull* for the object's mesh, and is represented by the Mesh Convex Hull option on the Simulation Geometry rollout of the Rigid Body Properties dialog.

However, there are times where you will want to simulate the exact mesh of an object, concavities and all. You specify this using the Concave Mesh option.

In this section of the tutorial, you will specify that you want to simulate your sphere as an exact sphere. This means that the sphere's geometry is not simulated; instead, a mathematical sphere is used for simulation. This is not only faster, but also more accurate (the object will roll perfectly smoothly) and less memory-intensive than a simulation based on a geometry-based sphere.

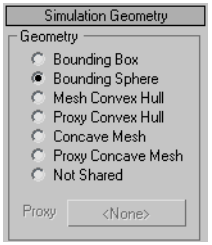
Continue using your scene from the previous lesson, or open *reactor_intro_3.max* from *\dynamics_and_effects\reactor\introduction*.

Change the simulation geometry for the sphere:

1 Select the sphere.

2  Open the Rigid Body Properties dialog.

3 In the Simulation Geometry rollout, choose Bounding Sphere.



4  Click Create Animation to regenerate the keyframes for the simulation.

The sphere rolls more smoothly, though this might not be immediately apparent if your sphere is highly tessellated.

5 At frame 0, select the sphere if necessary, and then Alt+click the sphere and, from the quad menu > Set quad, choose Delete Selected Animation.

The sphere's keys on the track bar go away.

Geometry Simulation

In this lesson, you'll continue making a toy. The existing sphere acts as the toy's body. The next step is to create a cylinder to act as the toy's arms. You'll also create a duplicate of this cylinder and reduce its tessellation. You'll use this simplified version, which is easier and faster to simulate, as the simulation geometry for the first, more complex cylinder. This technique is known as using *proxy geometry*.

NOTE If you were to make 20 copies of the first cylinder and add them to your simulation, they would all use the simplified, duplicate cylinder as their simulation geometry. This means you'd need only one instance of the cylinder geometry for the physical simulation, thus reducing memory usage and increasing simulation speed.

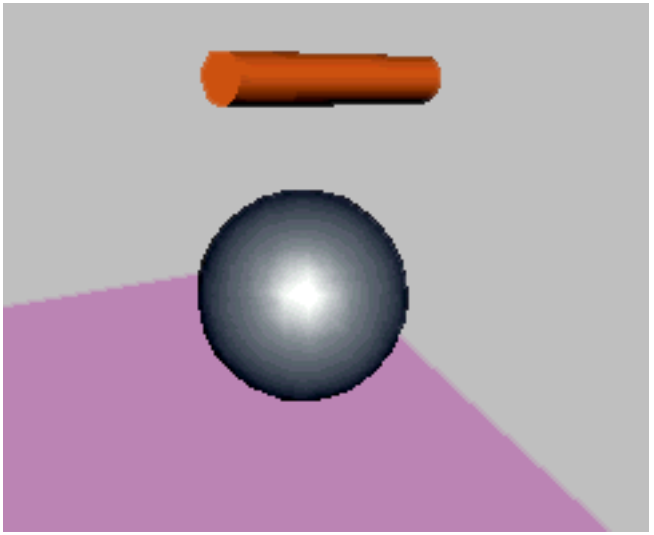
NOTE Continue using your scene from the previous lesson, or open *reactor_intro_4.max* from *\dynamics_and_effects\reactor\introduction*.

Create a simplified version of an object for simulation:

1 Create a cylinder in the Left viewport, and position it above your sphere.

Make sure that the cylinder and sphere don't touch each other.

2 On the Modify panel, set the following values for the cylinder: Radius=**6.0**, Height=**70.0**, and make sure Sides is set to 18 (the default).



3 With the cylinder still selected, choose Edit > Clone. On the Clone Options dialog, choose Copy and click OK.

You'll use this copy of the cylinder as the proxy geometry.

4 On the Modify panel, reduce the number of sides of the new cylinder to **12** and then move it away from the other objects.

5  Select the original cylinder and open the Rigid Body Properties dialog.

6 Set the cylinder's Mass value to **10.0**.

7 Set the cylinder's Simulation Geometry property to Proxy Convex Hull.

This means that the cylinder will use the convex hull created from another object's geometry as its physical representation.

The Proxy button at the bottom of the rollout becomes available.

8 On the Simulation Geometry rollout, click the Proxy button, and then select the cloned cylinder in one of the viewports.

The button now displays the name of your chosen proxy object.

TIP You can now hide your proxy cylinder to keep it out of the way by selecting it, right-clicking and choosing Hide Selection from the Display quadrant of the quad menu.

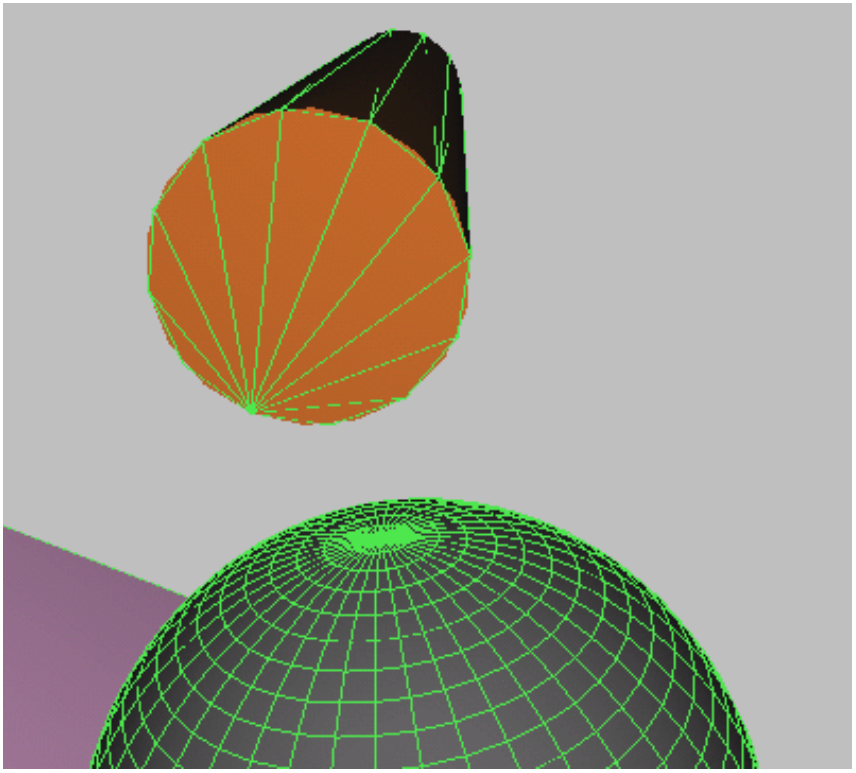
9 Select the RBCollection helper object in your scene, and add the original cylinder to it.

10  On the reactor toolbar, click Preview Animation and then play the animation.

The cylinder doesn't roll as smoothly as its display would suggest.

11 In the preview window, choose Display > Sim Edges to display edges for the objects' simulation geometries.

The cylinder's simulation geometry is coarser than its display body.



Building Up a Rigid Body

A rigid body can comprise more than one object, or primitive. A rigid body with more than one primitive is known as a compound rigid body.

As stated previously, concave objects (objects with holes or dips in the surface or geometry) are more difficult to simulate than convex objects. For this reason, a compound rigid body made up of several convex objects can be simulated much faster than a single concave object. A good example of this is a picture frame, where simulating a concave picture frame would be slow, but simulating a group of four convex boxes representing the frame sides would not.

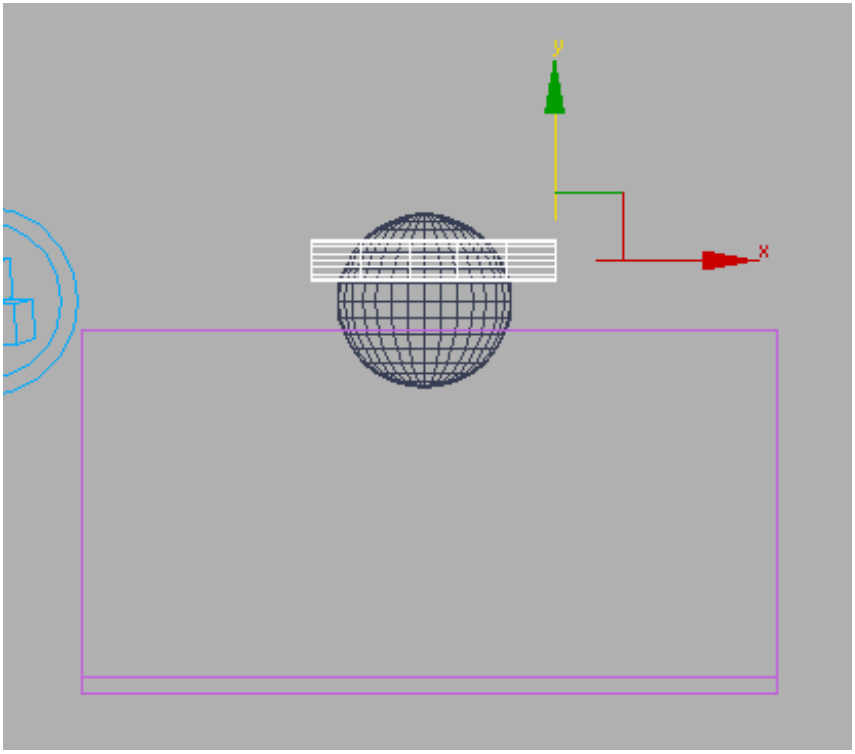
In this lesson, you will combine the two non-fixed rigid bodies in your scene (the sphere and cylinder) to create a single rigid body. In doing so, you will see the differences between primitive properties and rigid body properties. Each primitive in a rigid body can have a separate mass and simulation geometry, while friction, elasticity, and display body properties are applied to the entire rigid body.

To create your compound rigid body, you will need to select the objects and group them using the group functionality in 3ds Max. This group can be added to the rigid body collection as a new rigid body.

Continue using the scene from the previous lesson, or open *reactor_intro_5.max* from *\dynamics_and_effects\reactor\introduction*.

Create a compound rigid body:

1. Select the cylinder and move it down into the sphere, so that it resembles a pair of arms.



Relocated cylinder as seen in the Front viewport

2 Hold down Ctrl and select the sphere, adding it to your selection set.

The cylinder and sphere should be the only selected objects.

3 Choose Group menu > Group.

Use the Group dialog to name the new group **toy_body**. Click OK to continue.

4 Select the RB Collection helper object, and add your new group to the collection.

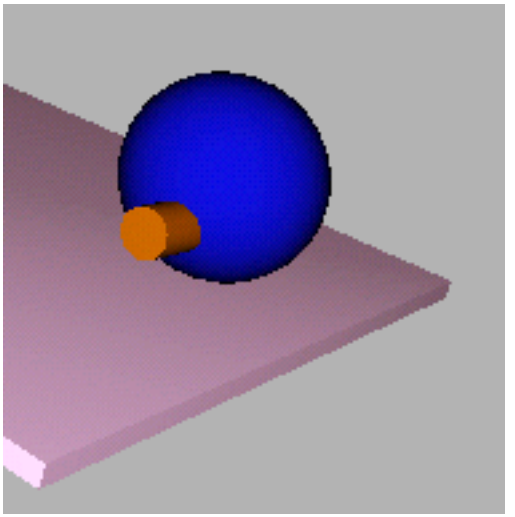
5  Click Preview Animation to watch the simulation.

You should get an error saying that the sphere and cylinder can no longer be rigid bodies. This is because they are being used as part of an active compound rigid body, so you will need to remove them as single objects from the rigid body collection.

6 On the RB Collection Properties rollout of the Modify panel, hold down Ctrl and select the sphere and the cylinder, and then click Delete.

7  Click Preview Animation and watch the simulation.

Now you can see that the sphere and cylinder act as parts of the same object.



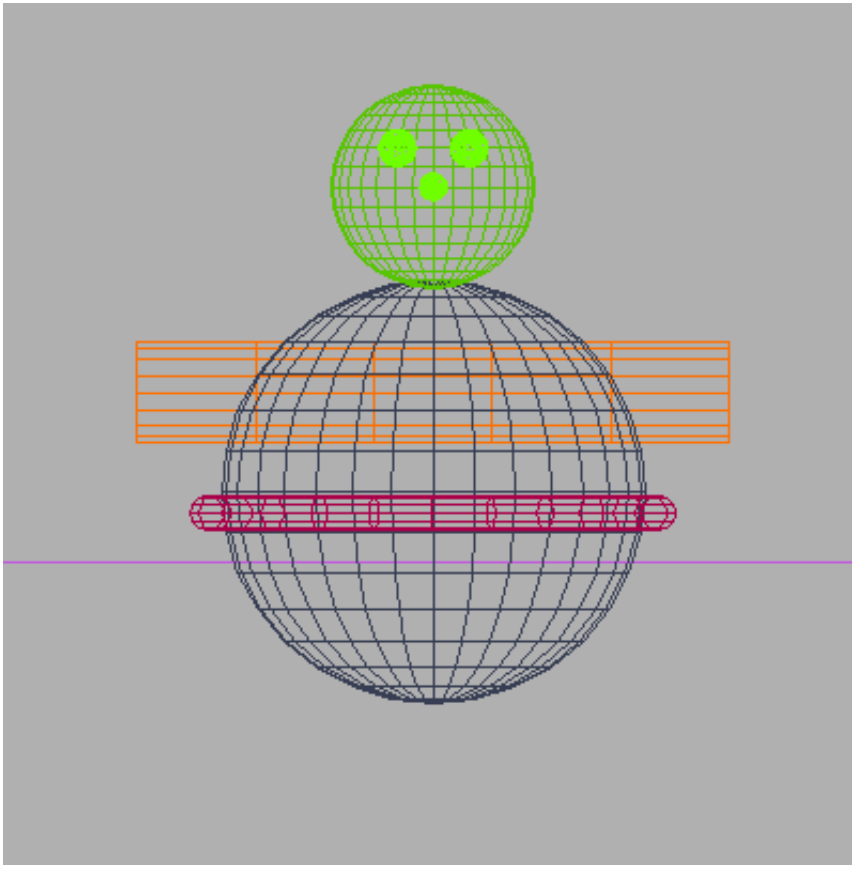
Making It Wobbly

The toy you are going to make will have a non-uniform mass distribution, which, in this case, means that most of its mass will be centered about its base. This will have the effect that the toy will not roll or fall over, but will try to right itself, and will wobble around. This behavior is possible because the primitives in a compound rigid body can have different masses. All you have to do is add a small, heavy object inside the bottom of the toy and most of its mass will reside there.

Continue using the scene from the previous lesson, or open *reactor_intro_6.max* from *\dynamics_and_effects\reactor\introduction*.

Complete the toy:

1 First, complete the toy's geometry. Create a torus for its waistband, a sphere for the head, two small spheres for eyes, and one for the nose.



Front view of completed toy geometry

2  Select the three spheres used for the eyes and nose and open the Rigid Body Properties dialog.

3 Set the simulation geometry to Bounding Box.

Since these spheres won't play a large part in the simulation, you can treat them as very simple geometry to speed up the simulation.

4 Select the head of the toy and set its simulation geometry to Bounding Sphere.

You can keep the torus's simulation geometry as Mesh Convex Hull, or you could create a copy and simplify it as a proxy, as you did with the cylinder.

5 Once you have finished creating and arranging these objects, add them to the **toy_body** group. For each object, select the object, choose Group menu > Attach, and then click **toy_body**.

Change the mass distribution for a rigid body:

1 Select the **toy_body** group and choose Group menu > Open.

This allows you to select the individual primitives within the group to edit their rigid body properties.

2 Change the mass of each object in the group (except the body and arms, which already have mass values) to **1.0**.

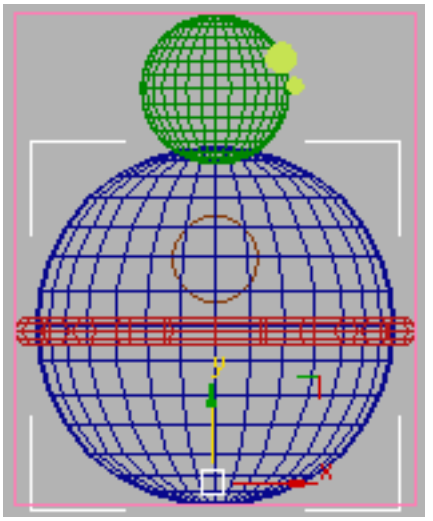
TIP You can do this quickly by selecting all of the objects in the group, and setting the Mass to **1.0** on the Rigid Body Properties dialog.

You do this because you are going to add a heavy object to the group, and it is this mass that you really want to affect the object's motion.

3  Click Preview Animation and you will see that all of the new objects have been added to the compound rigid body.

The toy should fall, though it won't yet have the correct behavior. If the object doesn't fall, you might not have given a mass to all of the objects in the rigid body and, as a result, one of the primitives is fixed. To fix this, close the preview window and check the mass of each of the objects in the group.

4 Add a small box (Height, Width, and Length=**3.0**) inside the body sphere and near the bottom of the object.

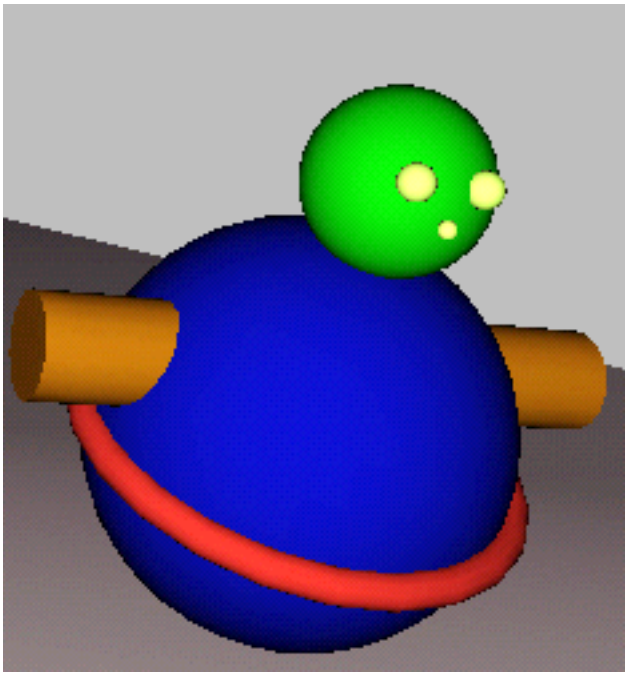


5  On the Rigid Body Properties dialog, give the box a mass of **300.0**.

6 Select a component of the group and choose Group menu > Close to close the group.

7 Select the small box, choose Group menu > Attach, and then click *toy_body* to add the box to the group.

8  Click Preview Animation and examine the behavior of the toy.



The toy slides down the box and tips over as before, but it tilts backward as though to right itself. The high mass in the box at the base of the toy has lowered the group's center of gravity.

Save your file as **my_reactor_intro.max**, and experiment with flattening out the plane and scaling it up.

Looking at Things Differently

reactor allows you to use *display proxies* for objects; this means that a rigid body can have a different display body for use in the preview window. This improves the window's performance, especially when you are simulating more than one object with the same display body. If you have a lot of objects that all look the same, then your display only needs to keep track of one instance of the display body when display proxies are used. As a bonus, you increase simulation setup speed, because you create only one instance of the display body.

In this lesson you'll create a display proxy for the toy. This is so that when you create copies of the toy later in the tutorial, your setup and display speed won't be adversely affected.

Continue using the scene from the previous lesson, or open *reactor_intro_7.max* from *\dynamics_and_effects\reactor\introduction*.

Create a display proxy for a rigid body:

1. Create a copy of the toy by Shift+dragging the group in the viewport.

   Move the copy away from the other objects.

2. A display proxy must be a single mesh. To create this, first ungroup the toy copy by selecting the new group and choosing Group menu > Ungroup. Then select the copy's main body sphere.

3. Right-click the sphere. In the Transform quadrant of the quad menu, choose Convert To > Convert To Editable Mesh.

4. In the Edit Geometry rollout of the Modify panel, click Attach List.

5. Use the Attach List dialog to select the rest of the objects that were part of the group, and then click Attach.

   You now have a single mesh representing the toy that you can use as a display proxy.

   TIP It is a good idea to label models in your scene clearly, especially when they look similar to one another. In this tutorial, for example, you could rename your proxy object **toy_body_proxy**.

6. Select your original **toy_body** group and choose Group menu > Open.

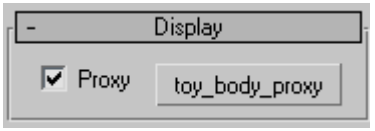
7. Select the group parent, which is the pink box surrounding the grouped objects.

   You need to select the group parent rather than any of the toy's constituent objects, because the group parent represents the rigid body and display proxies are applied to rigid bodies, not primitives.

8.  Open the Rigid Body Properties dialog, and on the Display rollout, turn on Proxy.

   This activates the display proxy pick button.

9. Click the display proxy pick button and select the single-mesh version of your toy in one of the viewports.



The button displays the name of your proxy object, in this case toy_body_proxy.

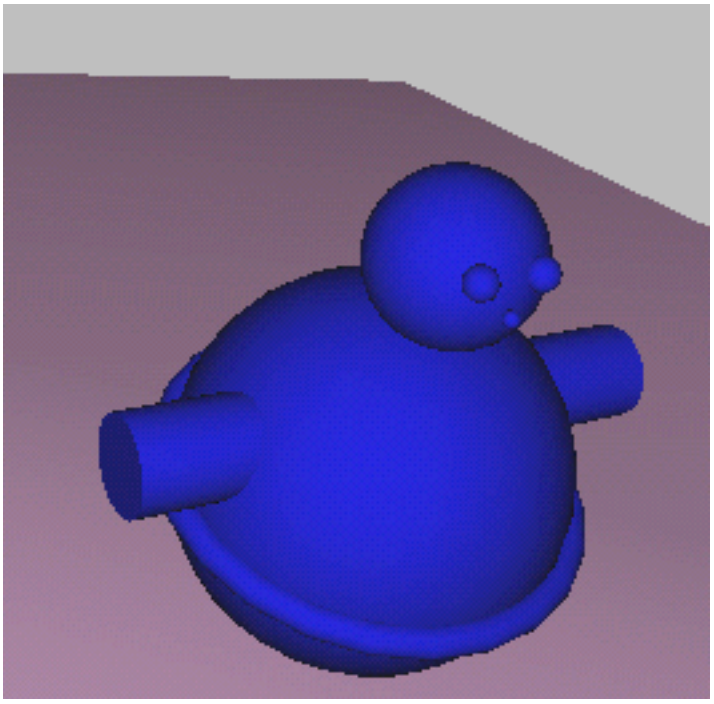
You have now assigned an alternative display body to your compound rigid body.

10 Hide the proxy object: Select the single-mesh toy, right-click, and from the Display quadrant of the quad menu choose Hide Selection.

This will keep it out of your way.

11  Select a component of the group, and choose Group menu > Close, then click Preview Animation.

Your toy will use the new mesh as its display object. If you make changes to the editable mesh, these will be displayed whenever you use the preview window.



Simulation Accuracy

If strange artifacts show up in the objects' motions, you probably need to increase the accuracy of your physical simulation. This can be done in one of two ways. You can change the accuracy externally, which affects the accuracy for both keyframe creation and the preview window, or you can alter it from within the preview window. In both cases, changing the accuracy requires altering a simulation parameter called Substeps/Keyframe. You can edit this in the **reactor** rollouts on the Utilities panel or by using the Physics menu in the preview window.

When the physics are being calculated, **reactor** moves the objects forward in small steps. The smaller the steps, the more accurate the simulation. However, you should note that the simulation will also become slower as it becomes more accurate. The default accuracy value is 10 substeps per keyframe, which means that **reactor** divides each keyframe interval into 10, and steps the simulation forward in intervals of this size. If you increase the substeps per

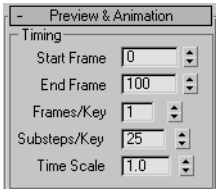
keyframe value to 25, then the size of **reactor**'s steps becomes one keyframe interval divided by 25.

NOTE You can continue using your file or you can open *reactor_intro_8.max* from *\dynamics_and_effects\reactor\introduction*.

Change the accuracy of your simulation:

1  On the Utilities panel, click the reactor button and expand the Preview & Animation rollout.

2 Change the Substeps/Key value from 10 to **25**.



3  Click Preview Animation.

The simulation may run more slowly but will be less likely to be affected by sudden slowdowns.

Setting Up the Simulation

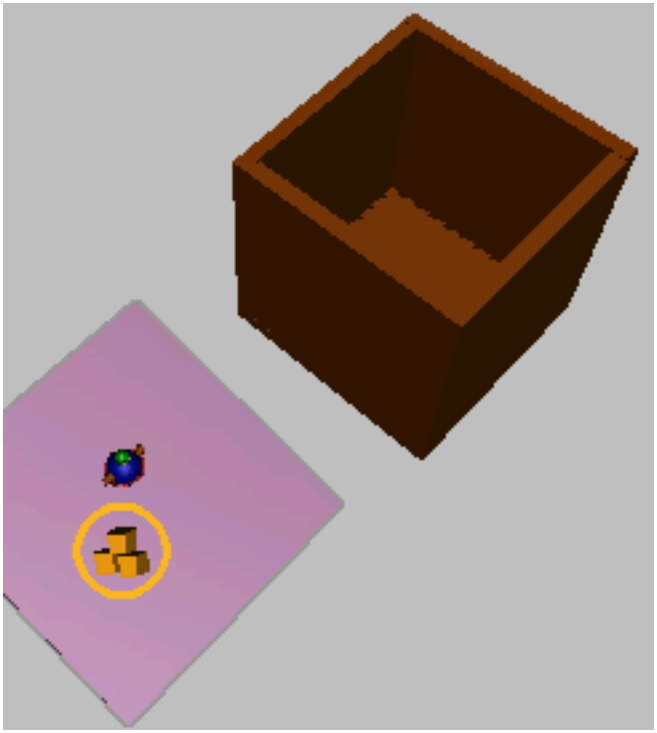
In this section you will create a toy box and 39 copies of your toy. Using the preview window, you will drop the toy copies into your box, let them settle, and then pass their positions back to 3ds Max. As you will see, the preview window acts as an interactive modeling tool, allowing you to rearrange objects physically and then use changes made in the preview window to update 3ds Max.

NOTE You can continue using your file or you can open *reactor_intro_9.max* from *\dynamics_and_effects\reactor\introduction*.

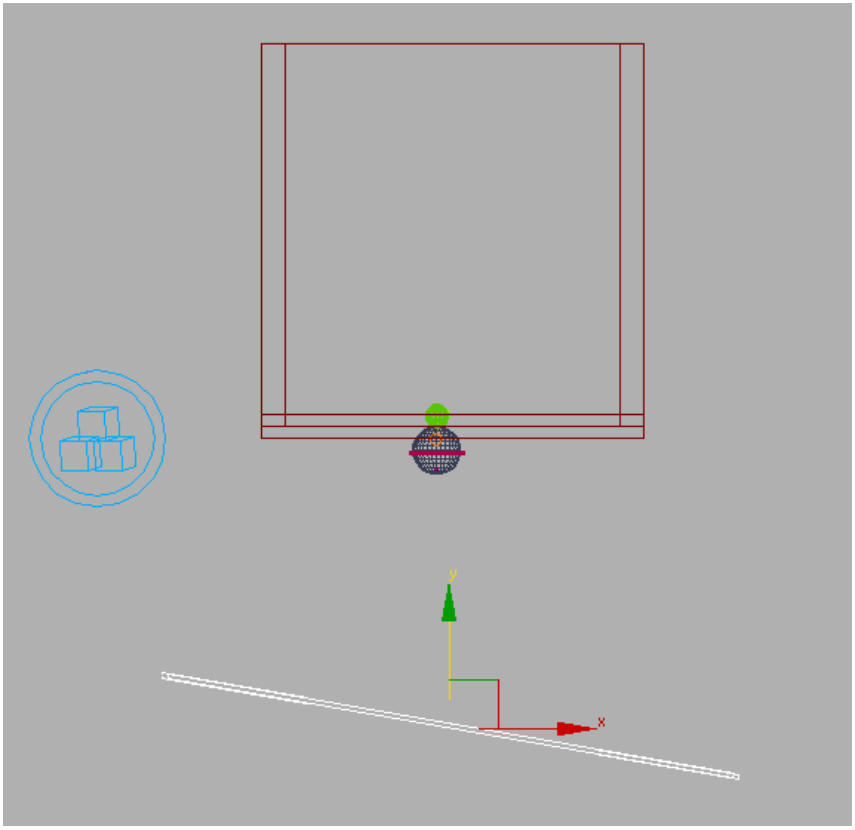
Create the toy box:

1 In your scene (away from the other objects), create five boxes with Length and Width=**400.0**, and Height=**25.0**.

2 Arrange the boxes to form a larger, hollow box.



3 In the Left viewport, rotate the sloping box 20 degrees counterclockwise to flatten it out somewhat. With the sloping box selected, increase both its Length and Width parameters to **600**, and move it downward 240 units.



4 Select and right-click the box used as the base of the toy box, and in the Transform quadrant of the quad menu, choose Convert To > Convert To Editable Mesh.

5 In the Edit Geometry rollout of the Modify panel, click Attach List.

6 In the Attach List dialog, select the other four boxes, which make up the sides of the toy box, and click Attach.

You now have a single mesh representing the toy box.

7 Open the Rigid Body Properties dialog and set the Simulation Geometry property to Concave Mesh.

Update your scene from the preview window:

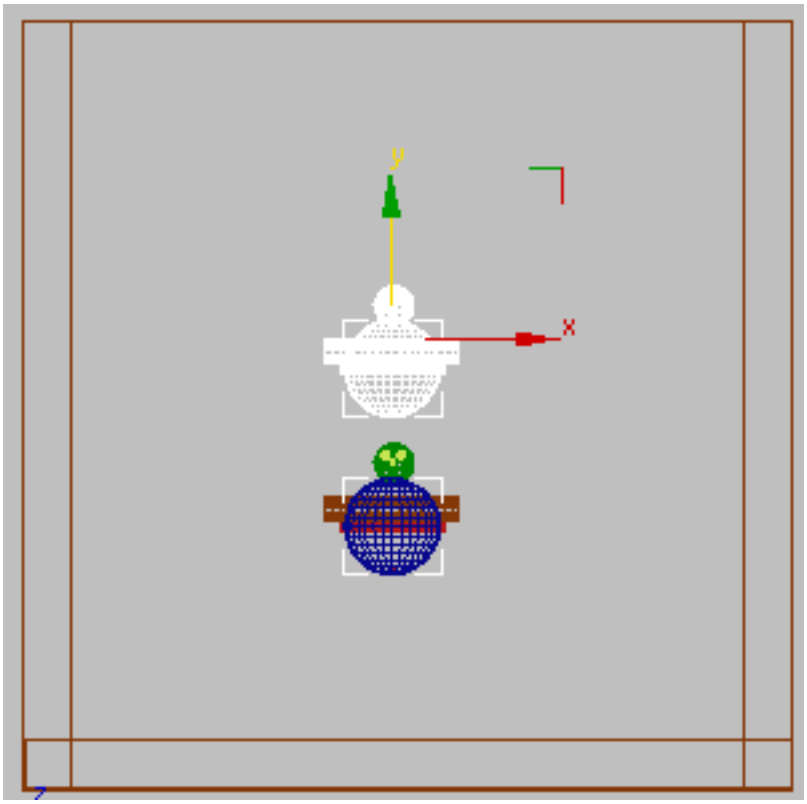
1  On the toolbar, click Select and Move.

2 Move the toy inside the box.

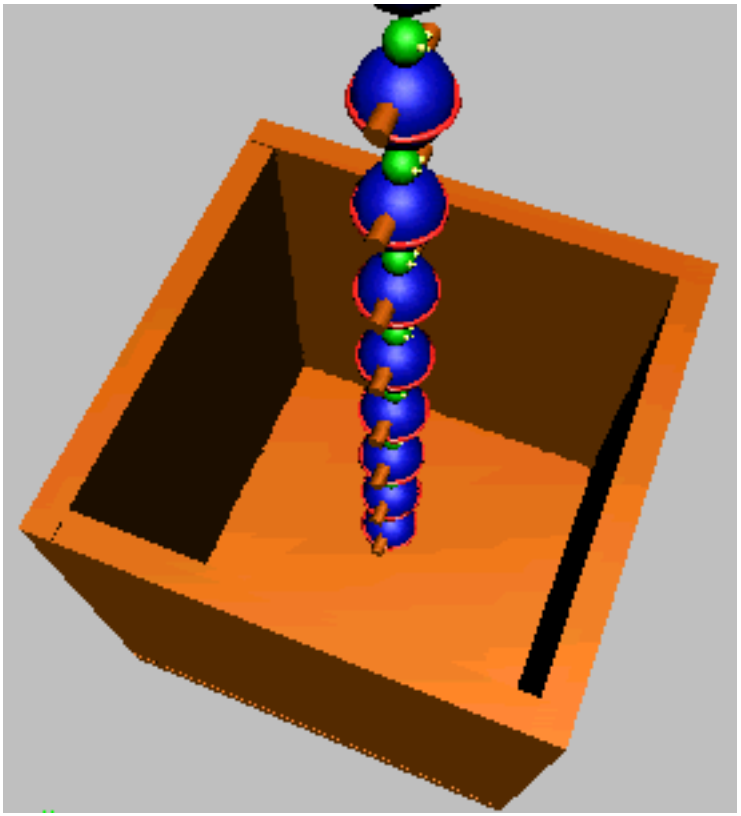
NOTE Make sure that the toy is not touching the box.

3 In a side viewport, Shift+Move the toy, and position the first copy above the original.

NOTE Make sure that the copy is not touching the original toy.



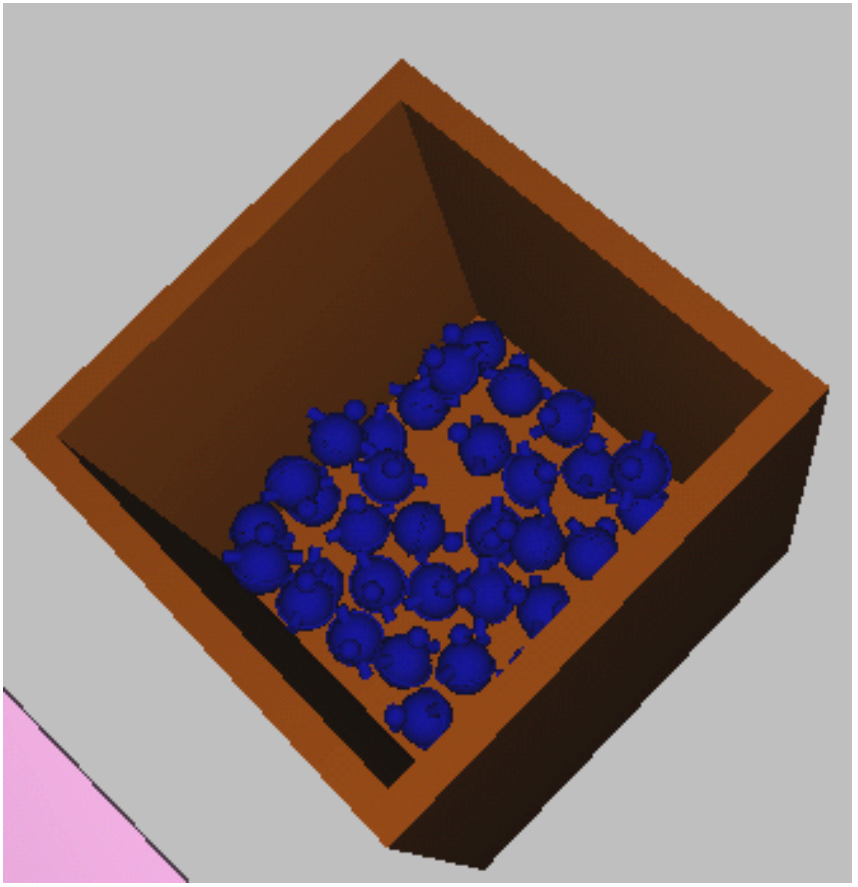
4 In the Clone Options dialog, set Number Of Copies=**39** and click OK.



5 Select the RB Collection helper object and in the Modify panel > RB Collection Properties rollout, click Add.

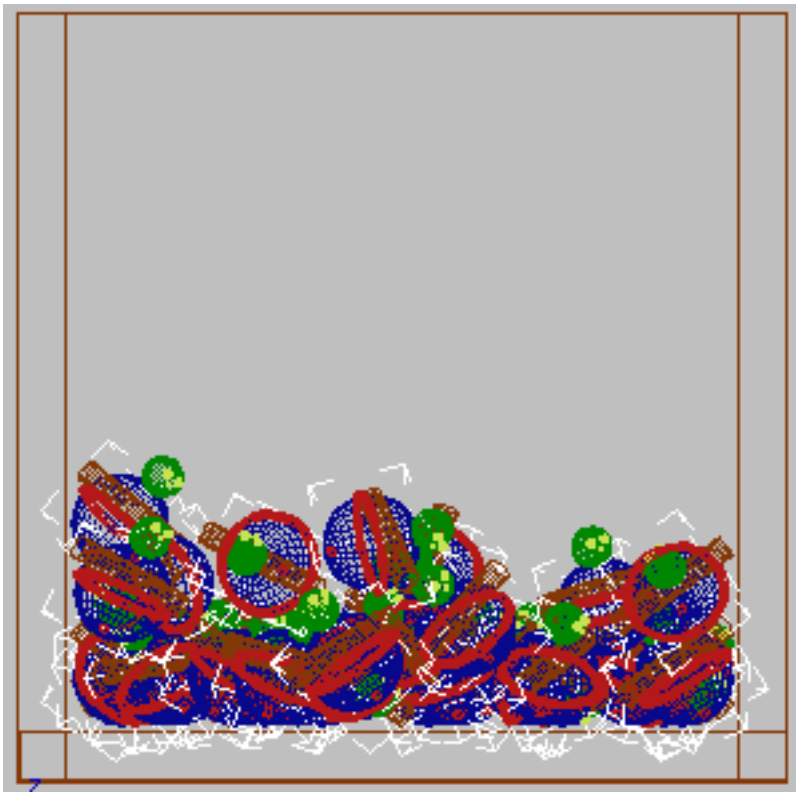
6 Select all of the new toy objects and the toy box and click Select to add them to the rigid body collection.

7  Click Preview Animation and let the toys fall into the box and settle.



8 When the toys have settled in the box, from the Preview Window menu bar choose MAX menu > Update MAX.

9 Close the preview window and click in one of the viewports to update the display. The toys will have been updated so that they have the positions and rotations you updated in the preview window.



Using Hand-Animated Rigid Bodies

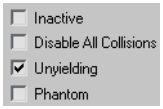
Something that you will often want to do is to use hand-animated rigid bodies in a physical simulation. These rigid bodies are referred to as *unyielding rigid bodies* and have the Unyielding property turned on in the Rigid Body Properties dialog. They behave as their keyframes dictate and other rigid bodies react to them. You do not need to give them a mass. In this example, you will hand-animate your toy box so that it tips over and spills the toys out onto the floor.

NOTE You can continue using your file or you can open *reactor_intro_10.max* from *\dynamics_and_effects\reactor\introduction*.

Add hand-animated rigid bodies to a physical simulation:

1. Select the toy box.

2.  Open the Rigid Body Properties dialog. On the Physical Properties rollout, turn on Unyielding.



3. 

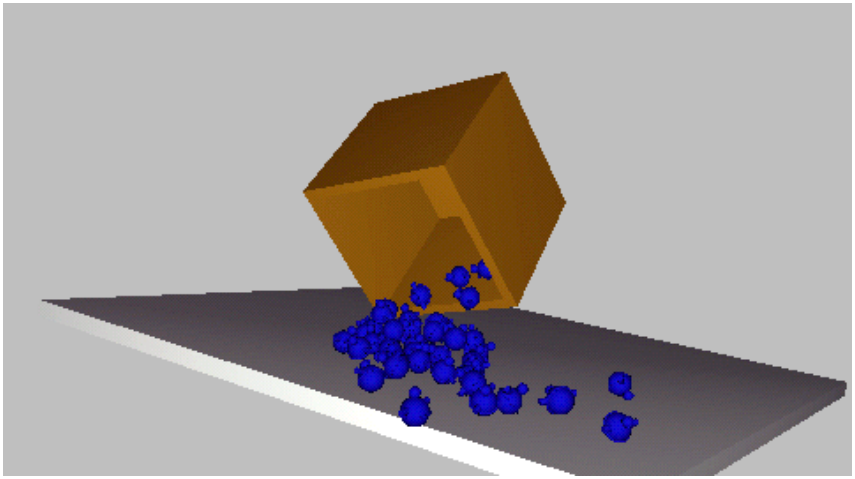
Turn on Auto Key and move the time slider to frame 60. Rotate the box clockwise by about 125 degrees so that the objects will pour out when physically active. Turn off Auto Key. Be careful to move your floor below the final position of the toy box so that the two are not interpenetrating. This would not generate errors in simulation, but would look strange.

NOTE Your toys may exhibit some strange behavior when you are hand-animating the box, but don't be alarmed. This is because the objects have not been keyframed yet. They will, however, behave correctly in the preview window.

---

4.  Click Preview Animation.

   Your box tips over and the toys pour out onto the floor. You may want to increase the size of your floor so that the toys don't all wobble over the edge.



5 At this stage, you may want to tweak the physical properties of your objects. For instance, if your objects are piling up as they fall out of the box, you can reduce their friction and the friction of the floor, so that they spread further after colliding with the floor and the other toys.

NOTE Remember that to change the physical properties, you must open the group and change the properties of the group helper object. You can change the properties of more than one object at a time.

Preparing Output

Now that you have your simulation behaving correctly, you need to create keyframes for the objects' motions. This can be done as before using the Create Animation button on the Toolbox.

In this example, you will create keyframes for the first 100 frames of the motion, and then simulate from frame 100 on. This shows how the **reactor** dynamics functionality can extract initial velocities from the scene so that simulating from frame 0 to 100 in one step is the same as simulating from frames 0 to 20, then from 20 to 40, 40 to 60, 60 to 80, and finally 80 to 100.

NOTE You can continue using your file or you can open *reactor_intro_11.max* from *\dynamics_and_effects\reactor\introduction*.

Extract initial velocities for rigid bodies:

1.  On the reactor toolbar, click Create Animation.

   Keyframes are created between frame 0 and 100 for your rigid bodies' motions.

2.  On the Utilities panel click reactor, then on the Preview & Animation rollout, set Start Frame=**100** and End Frame=**500**.

3.  Click Preview Animation.

   You will see that the objects start simulation with initial linear and angular velocities.

4.  In the Time Controls, click Time Configuration.

5. In the Time Configuration dialog, set End Time to **500**.

6.  Click Create Animation again.

   Keyframes will be created from frame 100 to 500. If you examine the keyframes behavior in the viewport, you will see that the objects' motion is smooth as it passes over frame 100, since the objects' initial velocities were extracted and used for simulation.

Reducing File Size

You now have keyframes for your objects' motion that can be used for creating animations or still renderings. However, each object currently has a keyframe for every frame. This means that your file size is quite large; it will also be quite difficult to tweak an object's behavior after the simulation.

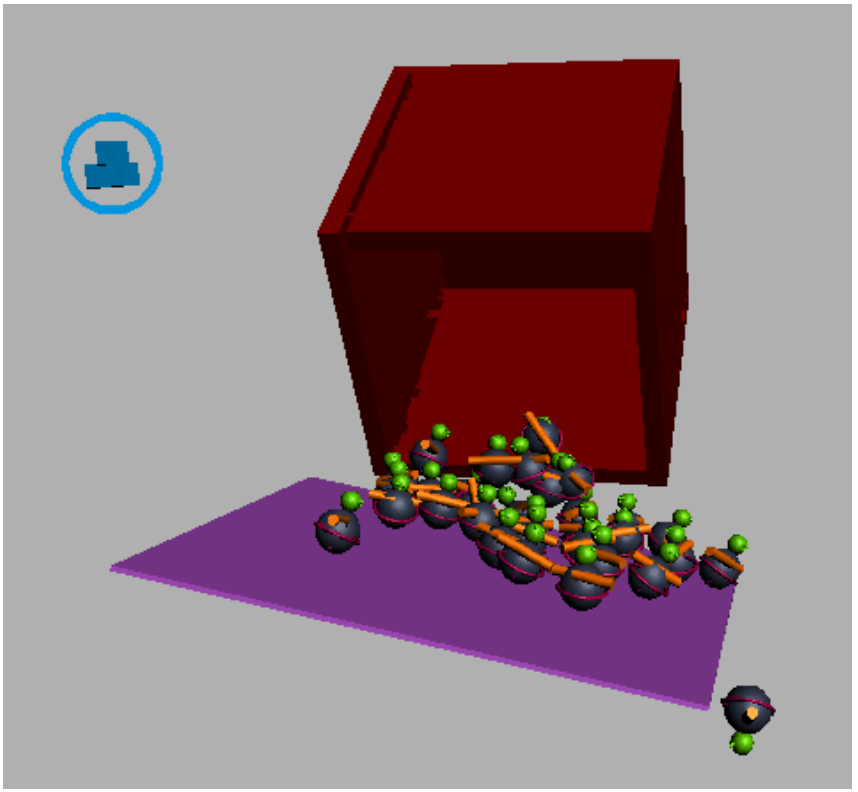
reactor provides access to a reduction process that attempts to remove redundant keyframes for the rigid bodies. You can provide a threshold value for the algorithm that guarantees that the motion of the objects after this reduction will be within that factor of their original value. In this section, you will reduce the keyframes for your rigid bodies.

NOTE You can continue using your file or you can open *reactor_intro_12.max* from *\dynamics_and_effects\reactor\introduction*.

Remove redundant keyframes after simulation:

1.  On the Utilities panel > click reactor and expand the Preview and Animation rollout.
2. In the Preview and Animation rollout, set Start Frame to **0** and End Frame to **100**.
3.  On the Utilities panel, expand the Utils rollout. In the Key Management group, click Reduce Now.

   This reduces the keyframes for all active rigid bodies in the simulation. For this example, use the default threshold value of 0.5.
4. Look at the animation again. As you can see, the behavior of the rigid bodies in your scene is almost exactly the same as before the reduction process, but the number of keyframes for each toy is greatly reduced. The motion of the toys is still quite complex, but the file size has been reduced about 30%.
5. Save your scene as *my_reactor_intro_final.max*. If you like, you can compare your results with the saved file *reactor_intro_final.max*. About midway through your animation, your scene should look something like the following illustration.



Summary

In this tutorial, you have learned the basics of using reactor to create complex animations:

1. Using the Rigid Body Properties dialog to set properties for objects in your animation.
2. Using an RB Collection object to control the animation.
3. Previewing the animation, and converting the animation into 3ds Max keyframes.

Rag Doll Tutorial

In this tutorial, you will add constraints to a predesigned character model to create a fully simulated rag doll using reactor. The tutorial uses two different constraint types to achieve realistic rag doll motion: Hinge and Ragdoll constraints. Each type is presented individually, with the workflow and typical use case for each constraint.

The scene is a simplified room, containing a set of stairs for your rag doll to fall down, and a floor on which it'll come to rest. The focus of this tutorial is rag doll character creation, so we've provided the completed scene with one catch: The character lacks any physical properties, and, most importantly, it has no constraints! Without these features the character has no *physical presence* in the scene, and will not react with the other objects.

As you proceed through this tutorial you will learn what constraints are, how to use them and how to tweak their parameters to adjust the rag doll's reactions. After this it is up to you to decide how to best use all those rag dolls, whether it be for falling down staircases, tumbling over banisters or simply being caught in explosions. It's up to you!

In this tutorial, you will learn how to:

- Set physical properties for the bones.

- View gravity and other simulation parameters.
- Create a Rigid Body Collection.
- Preview animations.
- Use constraints to hold the character together.
- Use Hinge constraints for knees, elbows and wrists.
- Disable collisions.
- Use Ragdoll constraints for hips and shoulders, back and neck.

Skill level: Intermediate

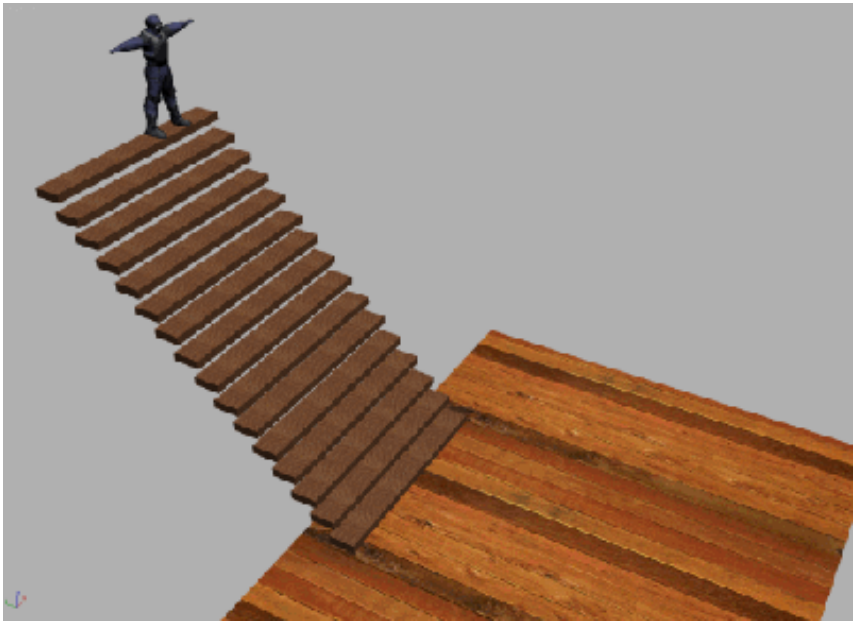
Time to complete: 2–3 hours

Setting Up the Rag-Doll Scene

In this lesson, you set up the scene by opening a 3ds Max file.

Open the sample file:

-  On the Quick Access toolbar, click the Open File button, navigate to the *\scenes\dynamics_and_effects\reactor\ragdoll* folder, then open *character.max*.
  A scene opens, containing a flight of stairs, a floor, and a character model.



At the moment, all objects in this scene include no physical properties; you will be adding these properties over the course of the tutorial.

reactor essentially adds *physicality* to scenes. To properly simulate dynamics, objects must have real-world properties such as mass and friction assigned to them. By adding these properties, you can create a simulation that mirrors your everyday experiences, or, if you'd prefer, an alien world with gravity pulling to the right!

Making Objects Physical

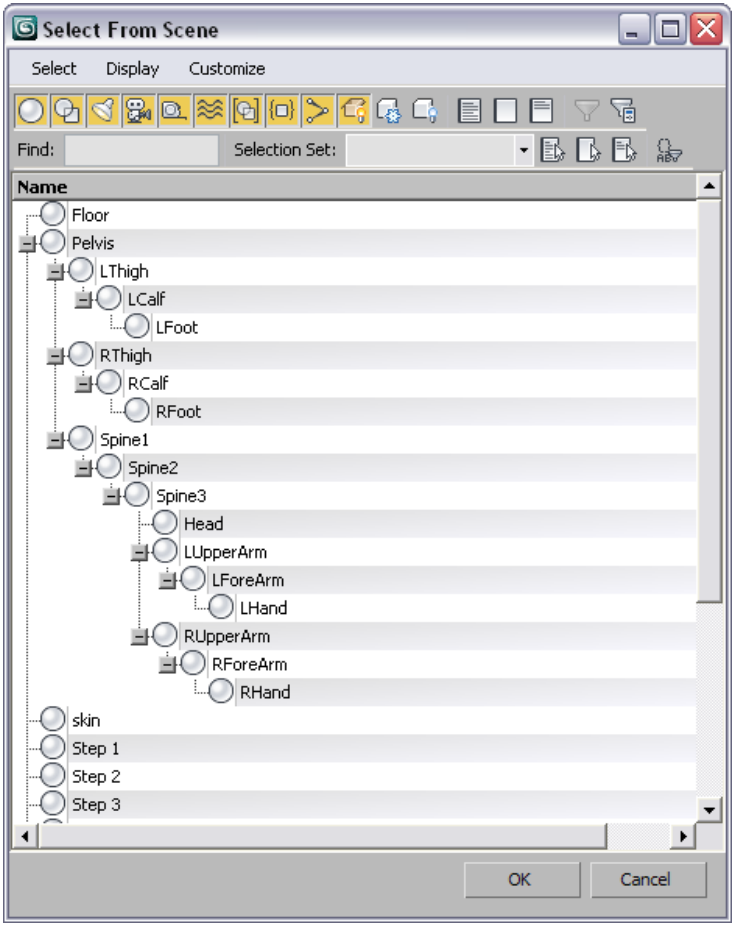
The character model you are working with may be thought of as 18 individual pieces: 17 of which are bones and an eighteenth the skin that envelopes them. As far as reactor is concerned, skin plays no part in the physical simulation of the scene; it is simply an aesthetic feature that masks the underlying bones. Only the bones themselves are simulated. They transmit coordinate information to the skin so that in turn may properly deform and keep the bones wrapped up.

With this in mind, the first thing you must do is give each of the bones a value for its mass.

Set physical properties for the bones:

1  On the toolbar, click Select By Name, or alternatively, press H. 3ds Max opens the Select From Scene dialog, which displays a hierarchical list of all the objects in the scene.

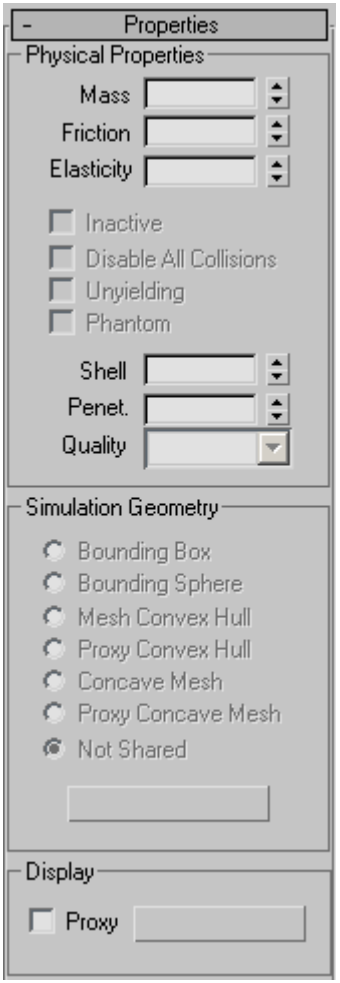
On the Display menu, make sure Display Children has a check mark by it, and then choose Display > Expand All.



As you can see, all of the bones are named so that their position in the body is apparent; **LCalf, RFoot, Head,** etc.

2 Select all 17 bones belonging to the Pelvis, LThigh and RThigh, and click OK.

3  On the Utilities panel > Utilities rollout, click reactor and open the Properties rollout.



4 In the Physical Properties group > Mass field, assign a value of **10.0** kilograms to each object in your selection. In total, the character will now weigh 170 kg. While this yields a nice final result, to add greater

realism to the simulation you can give each object its own value, comparable value to its real life equivalent. But be sure to keep the ratio of masses relative. For example, your forearm should not have the same mass as your thigh!

For this tutorial, set Friction and Elasticity values at 0.3. However, feel free to tweak these values and see the results. Elasticity controls how bouncy the object appears to be; a greater value leads to a bouncier feel. Friction controls how easy it is to slide the object across a given surface in the scene; larger values lead to increased resistance.

Next, you will give the background objects (the steps of the staircase and the floor) a physical presence in the scene.

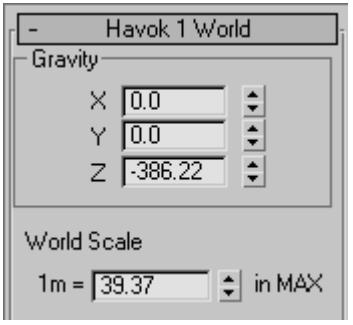
5 Select the steps and the floor in the scene.

6  On the Utilities panel > Properties rollout, note that Mass is to **0.0**.

This seems like a strange mass for these objects. However, reactor treats all objects with a mass of zero as fixed in place; under no circumstances are they allowed to move. Since you don't want the steps or the floor to suddenly drop into the void under the influence of gravity, you give them a mass of zero to tell reactor that they are immovable.

View gravity and other simulation parameters:

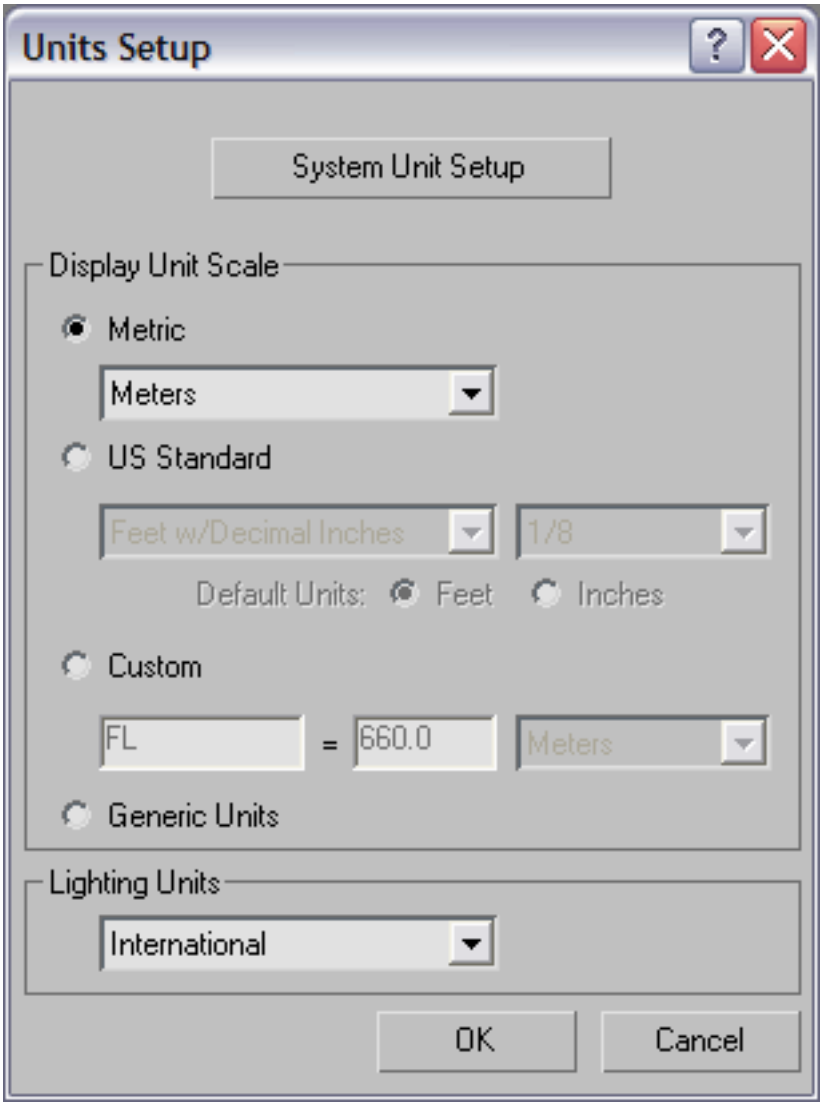
1 With the Utilities rollout > reactor still selected, open the Havok 1 World rollout.



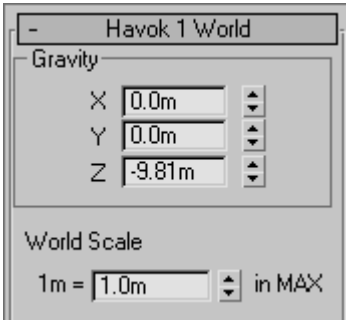
The three values in the World rollout > Gravity area control the gravity in your world (scene). The X, Y and Z components together form the direction vector that represents gravity. Since gravity usually points

downward, only the Z component has a non-zero value, by default. But where does the rather strange looking value of -386.22 come from? The minus sign simply indicates that the gravity should be applied downwards, and the value is arrived at as follows: In the real word, gravity accelerates objects towards the earth at around 9.8 m/s every second. If you look down at the next value in the World rollout, you'll see that 1m is equal to 39.37 3ds Max units (inches), so if you roughly convert gravity from meters to 3ds Max units, you get the 386.22 3ds Max units you see above.

2 Choose Customize menu > Units Setup and switch the Display Unit Scale to Metric.



Notice that the values in the Havok 1 World rollout are updated accordingly:

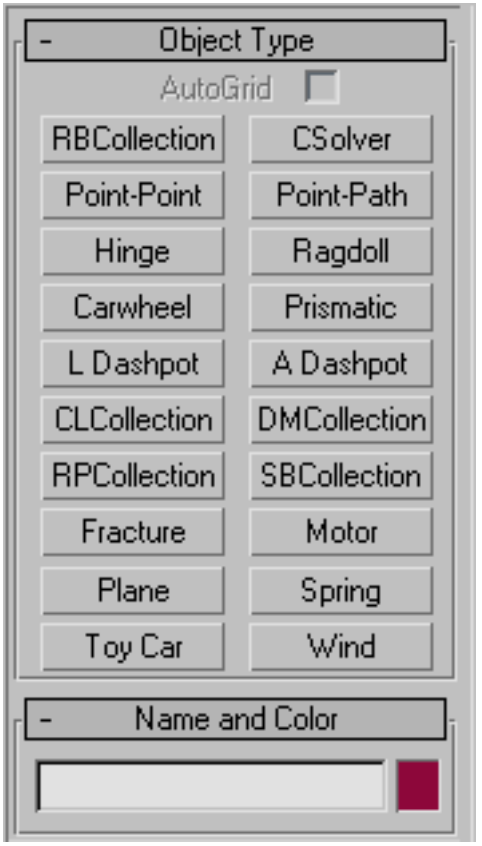


If you give values to the X and Y components of Gravity, you can create bizarre scenes where characters fly towards the walls or even the ceiling! In this lesson, you will be simulating a 'normal' environment, so there is no need to change it.

The final step before you can simulate your scene is to add your objects to a **rigid body collection**.

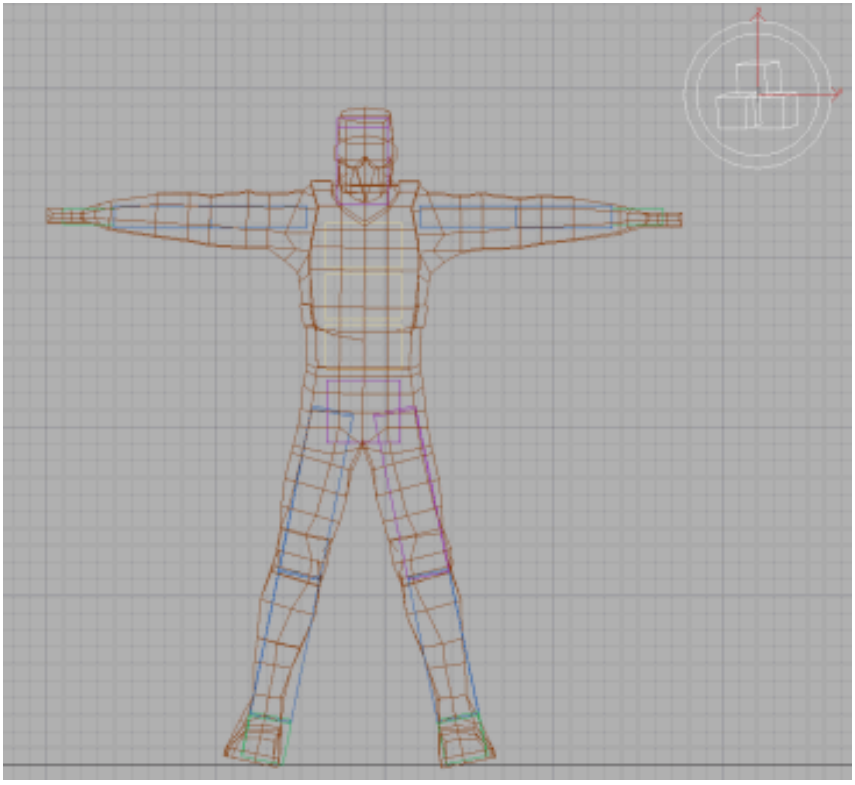
Add the objects to a rigid body collection:

1  In the Create panel, click Helpers and select reactor from the dropdown list.



2 On the Create panel > Object Type rollout, click RBCollection, then click in one of the viewports to create a rigid body collection helper object.

The position of the helper object is of no significance to the scene. You should see something similar to the following image in the viewport:



At the moment, there are no objects in the collection.

3  On the Modify panel > RB Collection Properties rollout, click Add.

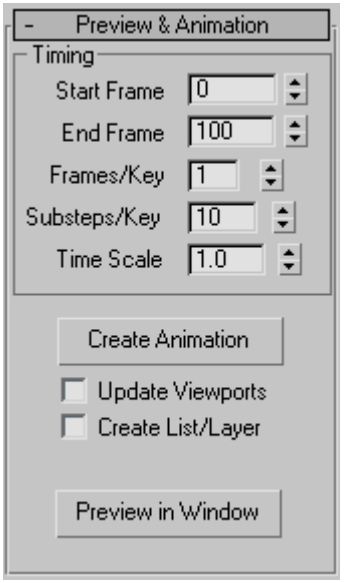
4 In the Select Rigid Bodies dialog, select all of the scene objects except *skin* and click Select.

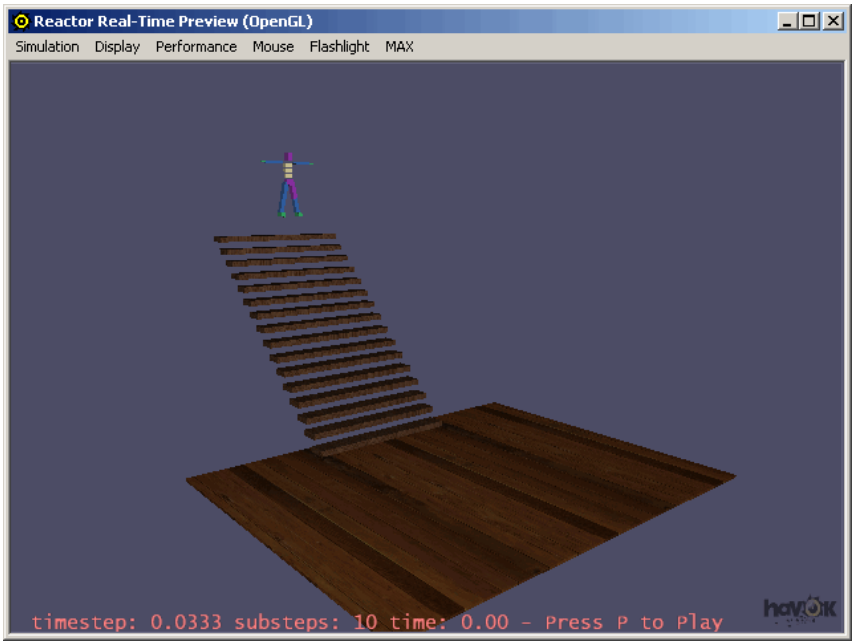
The RB Collection Properties list should now look something like this:

You're now ready to simulate the rag doll scene. Before you do, save your work as **my_character01.max**.

Preview the animation:

1. Continue from your previous section, or open *character01.max* from the *\dynamics_and_effects\reactor\ragdoll* directory.
2.  On the Utilities panel, click reactor and expand the Preview & Animation rollout, then click Preview in Window.





3 Press P to start the simulation.

The bones are not yet connected, and thus, fall apart when they hit the stairs. In the next lesson, you will attach the limbs.

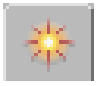
4 Close the Preview window.

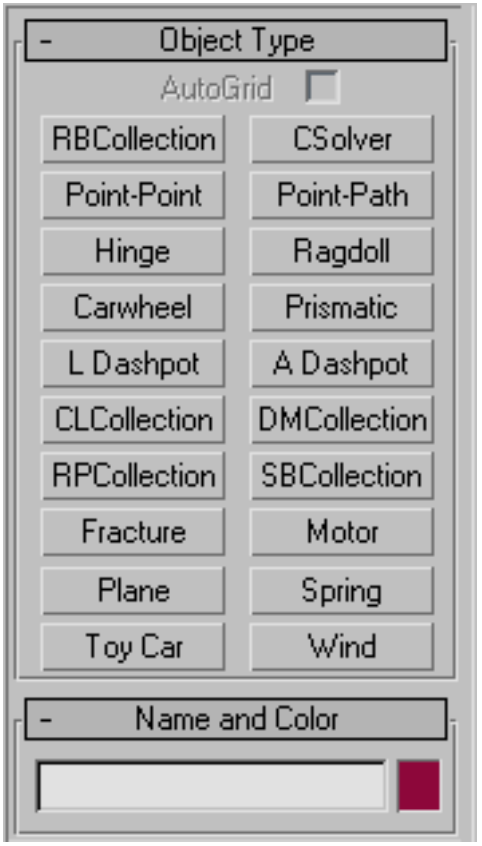
Attaching Limbs Together

In the previous lesson, the character literally fell to pieces once he hits the stairs. Each bone is essentially an independent object and is in no way linked with the other bones in the body—there is nothing to stop the individual bones from going their own way. In this lesson, you will use Constraints to hold the character together.

Constraints are used to connect one object to another. All constraints are represented in 3ds Max by helper objects.

View all available reactor helpers:

-  On the Create panel, click Helpers, and select reactor from the drop-down list.



Among the reactor objects you'll see some of the available constraint types; from Hinge constraints to Prismatic constraints, there are constraint types to reproduce pretty much any behavior you would like to see in your scene. In the next lesson, you will concentrate on the Hinge and Rag Doll constraints, as these are the two you will be using to piece together the character.

Using Hinge Constraints

The first constraint you'll be using is known as a **hinge** constraint which, as you might guess from the name, behaves like a hinge. A simple, everyday example of this kind of constraint is a door. A door can only swing around the points at which it is attached to the frame; you can think of this as the

axis around which the door rotates. It can't rotate around the other two axes, because the connection points to the frame prevent it from doing so.

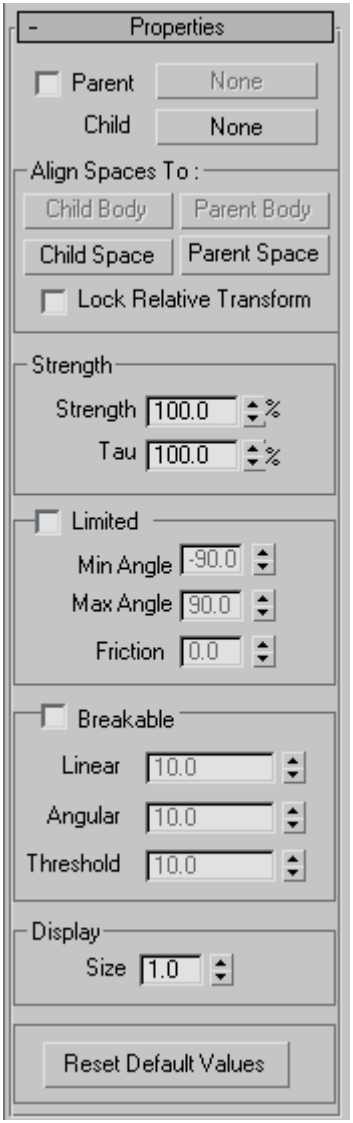
In fact, the door is actually a special type of hinge constraint in that it is a **limited** hinge constraint. This means that although it can rotate around the hinge axis, it can do so only until it bumps in the wall of the car. In reality, it cannot swing a full 360 degrees: The constraint's range of motion is limited (to 120 degrees, for example).

Furthermore, the hinges that connect the door to the frame can take only a measured amount of stress before giving way (think of hitting the door with a rather large sledge hammer). Since you can break the constraint, you would call this a **breakable** hinge constraint, or more precisely a **breakable limited hinge** constraint. The good news is that reactor can simulate all of these features, giving you free rein over the movement and the stresses and strains that the constraints in your scene can undergo.

Add a hinge constraint

1.  On the Create panel, click Helpers, and choose reactor from the dropdown list.
2. On the Create panel > Object Type rollout, click Hinge.

   The Properties rollout displays.



As you can see, Hinge has quite a few parameters, but the only ones that you will be concerned with for this tutorial are the Parent, Child, and Limited settings. You use the Parent and Child controls to specify the objects in your scene that serve as the two components of the constraint; for example: the upper arm and forearm. The Limited group parameters let you control the extents of the hinge's motion.

Setting Up Knees, Elbows, Ankles and Wrists

Open the sample file:

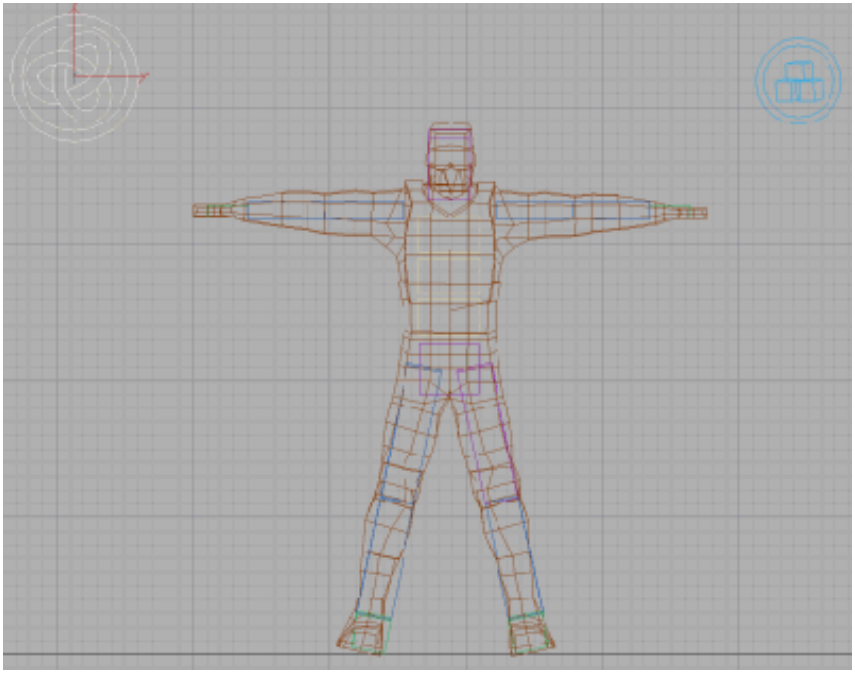
- Continue from the previous lesson or load the *character01.max* from the *\dynamics_and_effects\reactor\ragdoll* directory.

Using a constraint is similar to the Rigid Body Collection, but this time you add a **Constraint Solver** to the scene. Next you can begin to add various constraint types to your objects and then add them to this solver. The solver, in essence, figures out how best to keep all of your objects happy under the conditions they've been given (body A connected to body B, body B connected to body C, etc.).

Add a Constraint Solver to the scene:

1.  On the Create panel, click Helpers and choose reactor from the drop-down list.
2. On the Create panel > Object Type rollout, click CSolver.
3. Click in one of the viewports to create a Constraint Solver.

NOTE The location of the helper object is not important.



NOTE The Constraint Solver works only if all the constraints act upon rigid bodies that belong to the same Rigid Body Collection.

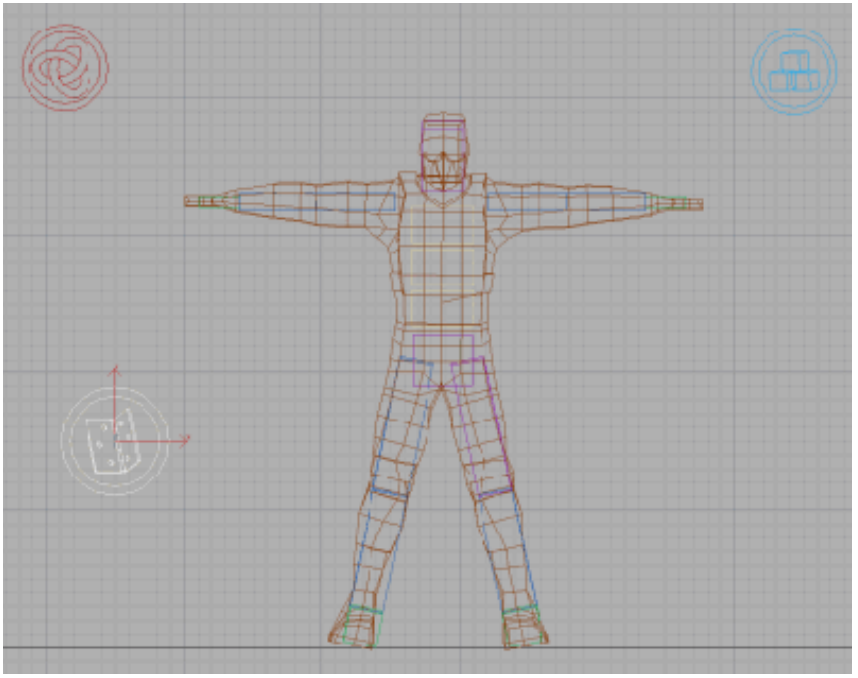
Next, you'll specify the collection to which to apply the Constraint Solver.

4. With the solver selected, open the Modify panel > Properties rollout, and click the button labeled None, below RB Collection.
5. Press H, choose **RBCollection01** from the Pick Object dialog, then click Pick.

Now you're ready to start connecting bones together, and soon will have your rag doll tumbling down the stairs. You'll use the Hinge constraint to connect the various bones in the character, first joining the right foot and calf bones to simulate an ankle.

Create Hinge constraints for the ankles:

1. On the Create panel > Object Type rollout, click Hinge.
2. Click anywhere in a viewport to create a Hinge constraint.



Next you'll choose the objects to which to assign to this constraint.

3 In the Modify panel > Properties rollout, click the button labeled None, next to Child.

4 Press H, choose **RFoot** in the Pick Object dialog, then click Pick

The Child button should now read **RFoot**.

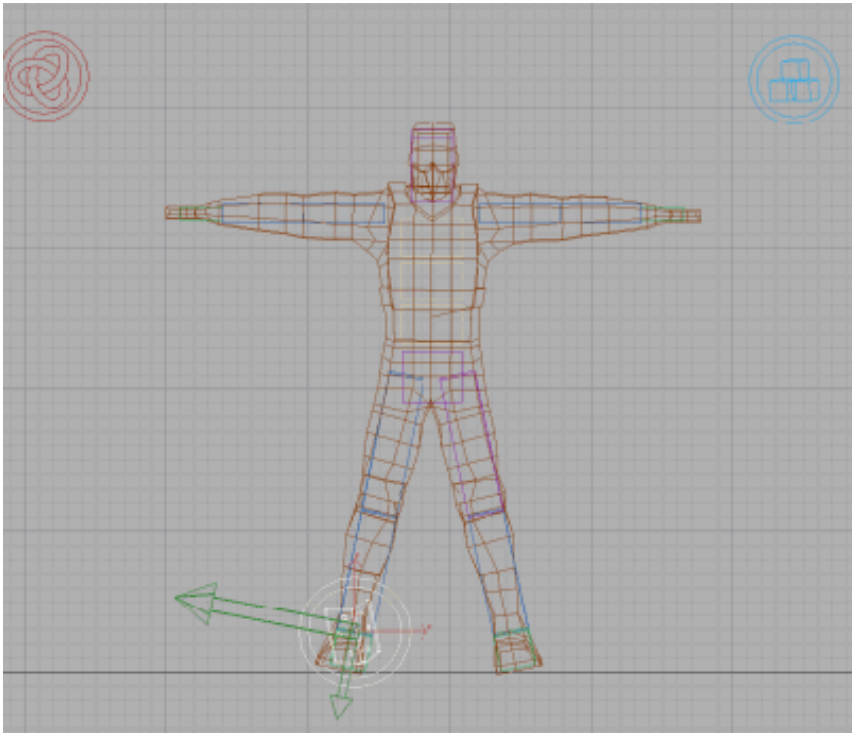
5 On the Modify panel > Properties rollout, turn on the Parent check box.

This indicates that you want to constrain two objects together, rather than one object to the world.

6 Click the button labeled None, next to Parent.

7 Press H, choose **RCalf** from the Pick Objects dialog, then click Pick

The button beside Parent should now read **RCalf** and the scene should look like this:

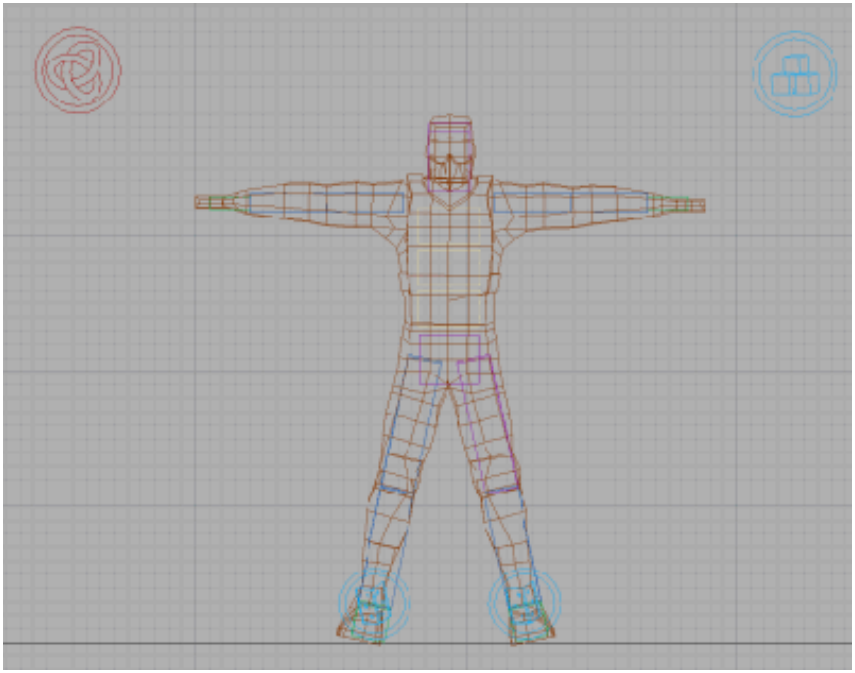


In its current state, the character's foot is free to rotate 360 degrees around the hinge constraint, which as you can imagine would be pretty painful! You need to use the Limited property for this hinge constraint.

8 On the Properties rollout, turn on Limited.

9 In the Limited group, set Min Angle to **–20.0** and Max Angle to **30.0**

10 Repeat steps 1 to 7 for the left foot and calf bones.



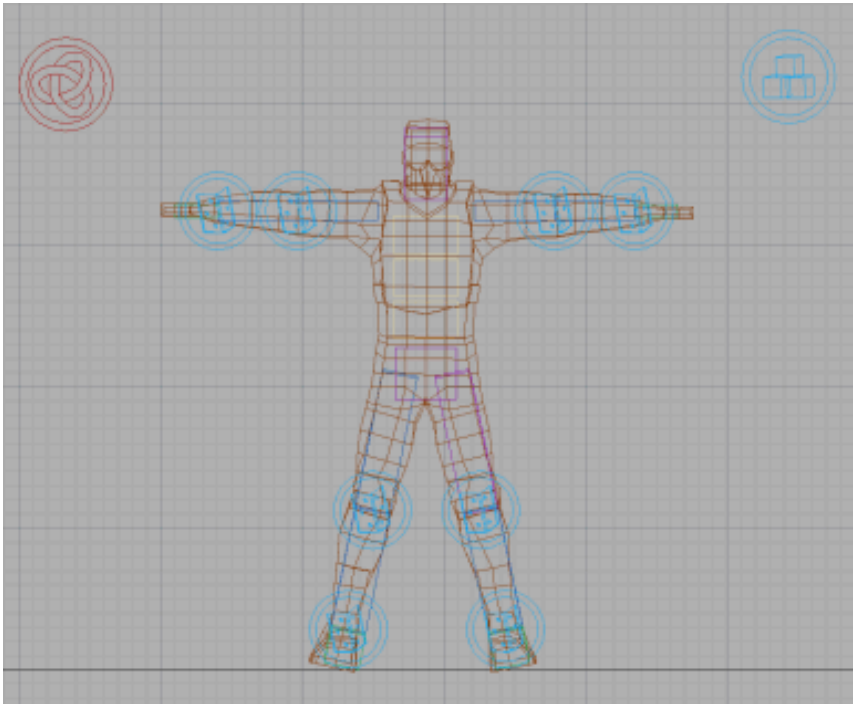
Create Hinge constraints for the knees, elbows and wrists:

- The process for setting up the hinge constraint for each of the other bone pairs is identical, with only the constraint limits differing. With this in mind use this list to set the constraint parameters for each joint:

Constraint Name	Child	Parent	Min Angle	Max Angle
RAnkle-Joint	RFoot	RCalf	–20.0	30.0
LAnkle-Joint	LFoot	LCalf	–20.0	30.0
RKnee-Joint	RCalf	RThigh	–85.0	0.0

Constraint Name	Child	Parent	Min Angle	Max Angle
LKnee-Joint	LCalf	LThigh	–85.0	0.0
RWrist-Joint	RHand	RFore-Arm	–35.0	35.0
LWrist-Joint	LHand	LFore-Arm	–35.0	35.0
RElbow-Joint	RFore-Arm	RUpper-Arm	–85.0	0.0
LElbow-Joint	LFore-Arm	LUpper-Arm	–85.0	0.0

When you're done, the scene should look like this:



You'll simulate the other bones using **Rag Doll Constraints**, as their motion is more sophisticated. The last thing you have to do is to add all of the constraints to the Constraint Solver.

Add the Hinge constraints to the Constraint Solver:

1. Press H, choose **CSolver01** from the Select From Scene dialog, then click OK.
2.  On the Modify panel > Properties rollout, click Add.
3. On the dialog, highlight all of the hinges and click Select

   The CSolver Properties rollout should now look like this:


Properties
RB Collection
RBCollection01
Constraints
highlight
LAnkleJoint
LElbowJoint
LKneeJoint
LWristJoint
RAnkleJoint
RElbowJoint
RKneeJoint
RWristJoint
Pick
Add
Delete
Deactivation Threshold
10.0
Disabled
Reset Default Values

Next you'll preview the scene again to see just how these alterations have affected the character.

-  On the Utilities panel, click reactor, and then on the Preview & Animation rollout, click the Preview in Window.

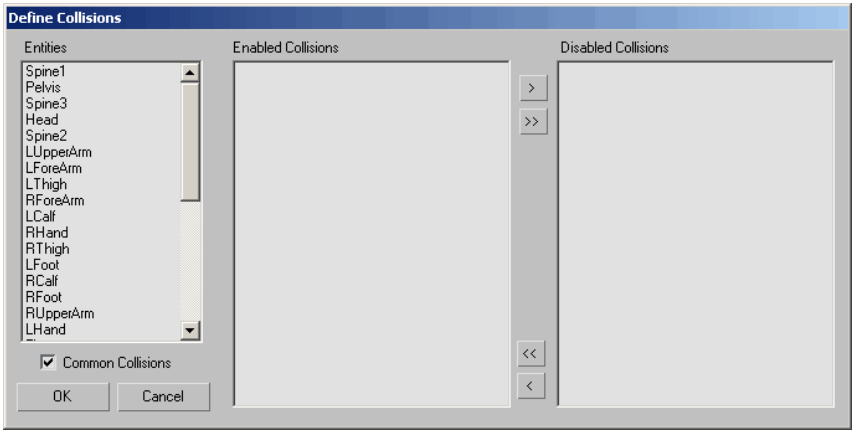
You might have noticed during the last preview that reactor complained that some of the rigid bodies in the scene were interpenetrating. Although you ignored this warning on that occasion, it's not a good idea to do in general. Later in this tutorial, you will take care of this issue, but for now click Continue and view the preview.

The first thing that stands out is that the bones connected with hinge constraints stay together. This is exactly as planned, but notice that they continue bouncing and dancing about once they hit the floor. This is because reactor notices that the bones are interpenetrating with one another and tries to separate them. However, at the same time, the Constraint Solver is trying to keep them together. Since they are essentially battling with one another over the position of the various bones they continue to bounce about erratically. The solution to this is to tell reactor not to worry about interpenetrations between the bones that you've connected.

Disable collisions between the limbs:

1.  On the Utilities panel > reactor > Collisions rollout > Global Collisions group, click Define Collision Pairs.

You are presented with the following dialog:

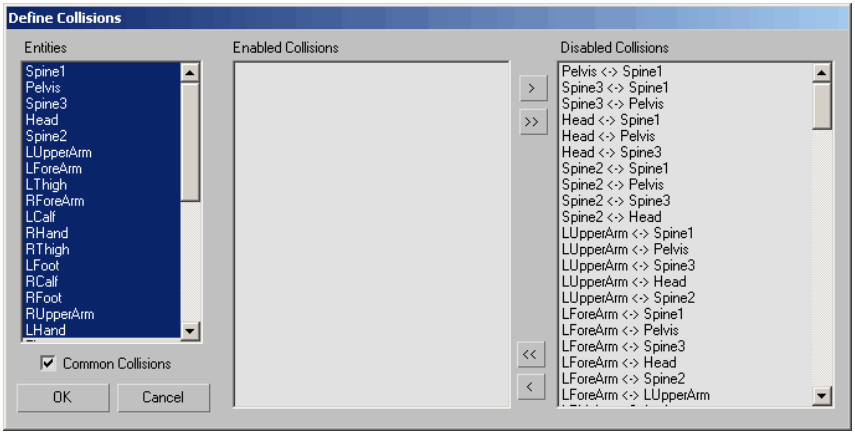


There are three columns: one for all the entities (objects) in the scene, one for enabled collisions, and one for disabled collisions. By default, all collisions are enabled between all objects so you need to tell reactor to ignore the collisions between the bones in your character. You want to ignore only bone-to-bone collisions, though. If, for example, you chose to ignore the collision between the character's foot bone and the floor object, the character's foot would pass straight through the floor unimpeded.

2 Select all of (and only) the bone objects in the first column.

This displays a list of enabled collisions in the second column.

3 Click the >> button



All of the collision pairs between the selected bodies move to the third column (disabled collision pairs).

4 Click OK to accept the changes.

NOTE Instead of disabling all collisions between all limbs, you could instead disable collisions between adjoining limbs. In order to do so, you would start with all collisions enabled, select each adjoining pair (**LHand** and **LForeArm**, **Spine3** and **Head**, etc.) and click ">>" to disable just those pairs.

Preview the animation with collisions disabled:

1  On the Utilities panel > reactor > Preview & Animation rollout, click Preview in Window.

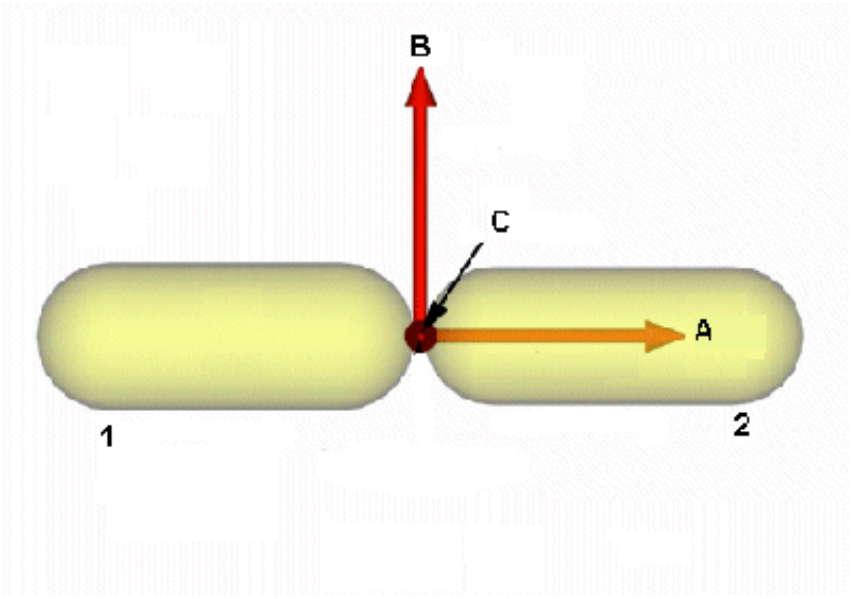
Press P to view the animation.

2 Save your work again, this time as **my_character02.max**

Using Rag Doll Constraints

In the previous lesson, you finished joining up the simpler bones in the body. Now it's time to think about how to simulate joints such as the hips or shoulders. These joints are more complex than the knees or elbow, as they need to allow for rotation and they have a range of motion of greater freedom than the hinge constraint provides. To tackle these types of joints you will use **Rag Doll constraints**.

These are fairly complex, so this lesson will serve as a brief overview. The following is a picture of two objects connected by a Rag Doll constraint:



A. Twist Axis

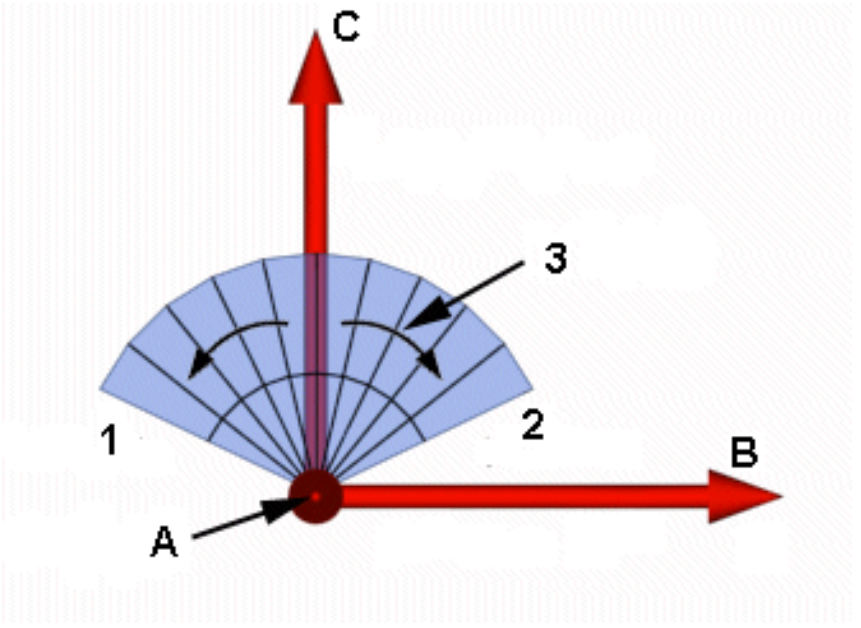
B. Plane Axis

C. Twist x Plane Axis (outward)

1. Parent Body

2. Child Body

The Rag Doll constraint provides three axes of importance: the **Twist Axis**, the **Plane Axis**, and the **Twist x Plane Axis**. The twist axis is probably the easiest to visualize and you can do so by simply stretching out your arm and rotating it so that your thumb changes from pointing upwards to downwards. Your movement is just a series of rotations along the twist axis of the shoulder and elbow. In general, the twist axis should follow the length of the child body in a joint. In reactor you can specify how far on either side of neutral the constraint can twist; this need not be symmetric.



A. Twist Axis

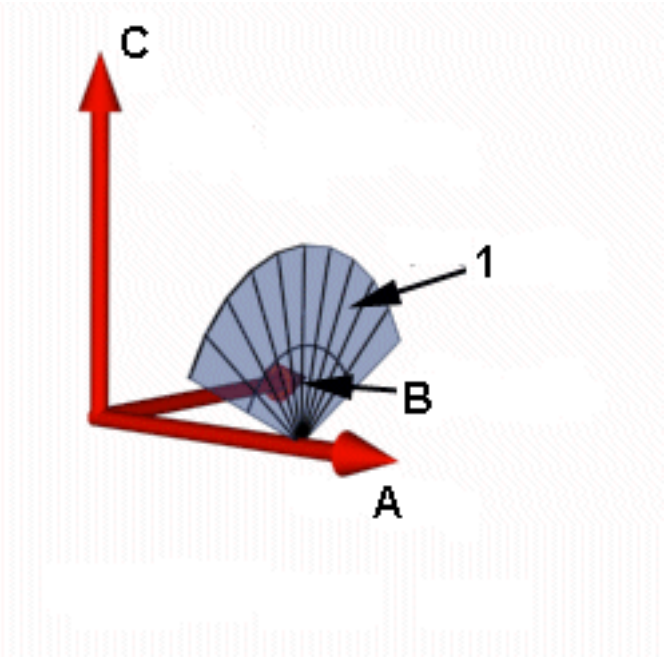
B. Plane Axis

C. Twist x Plane Axis (outward)

1. Twist Min.

2. Twist Max.

3. Twist Range



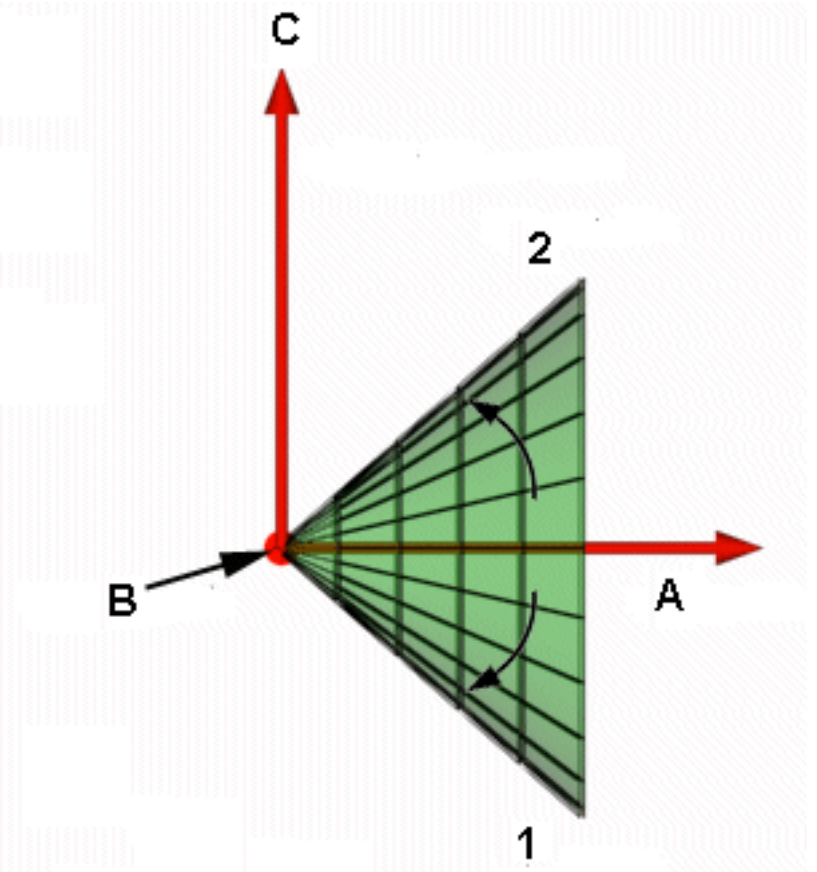
A. Twist Axis

B. Plane Axis

C. Twist x Plane Axis (outward)

1. Twist Range

The other two axes are interlinked and together the control the allowed volume that the child body can occupy. So first you'll have a look at the **cone** angles that allow you to specify the volume that the child may move through:



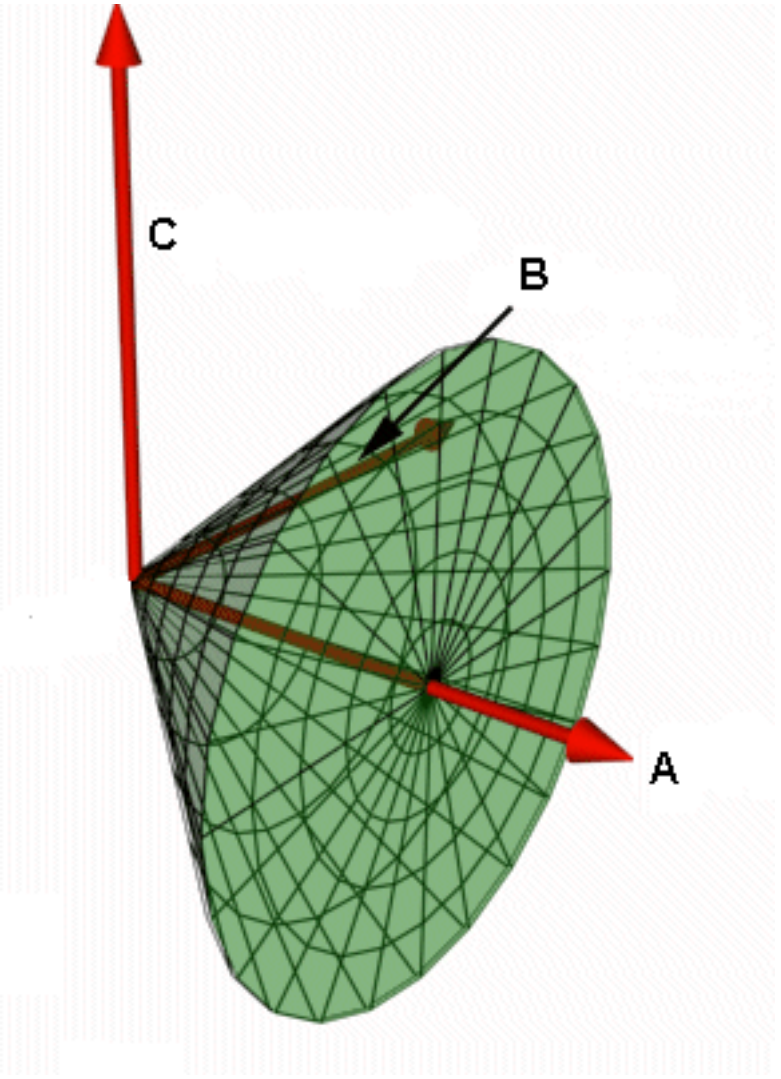
A. Twist Axis

B. Plane Axis

C. Twist x Plane Axis (outward)

1. Cone Min.

2. Cone Max.

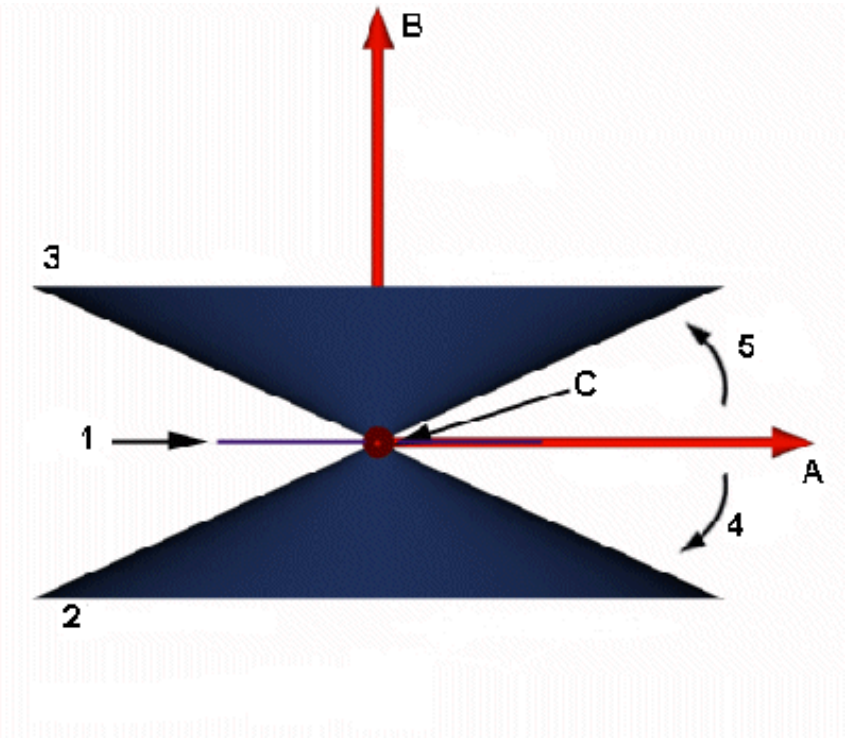


A. Twist Axis

B. Plane Axis

C. Twist x Plane Axis (outward)

If you imagine the constraint representing your shoulder, the green volume is the range of allowed positions that your upper arm may occupy. However, you can refine the allowed volume further by using the plane axis to enforce **plane limits**. These limits are used to generate two further cones:



A. Twist Axis

B. Plane Axis

C. Twist x Plane Axis (outward)

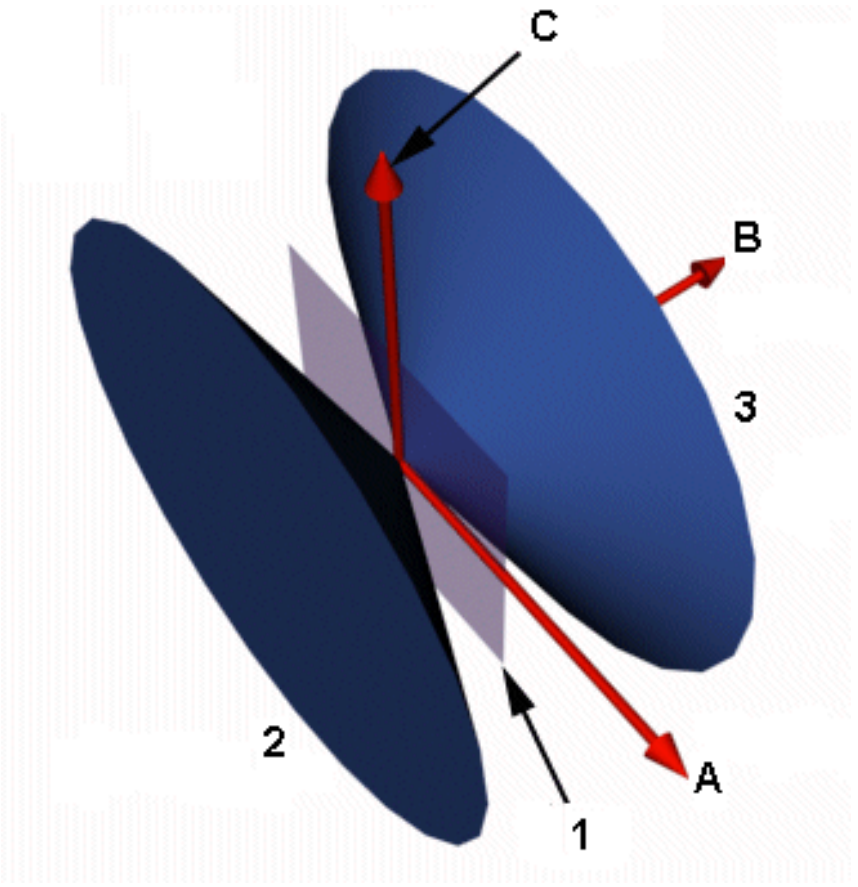
1. Plane

2. Plane Min. Cone

3. Plane Max. Cone

4. Plane Min.

5. Plane Max.



A. Twist Axis

B. Plane Axis

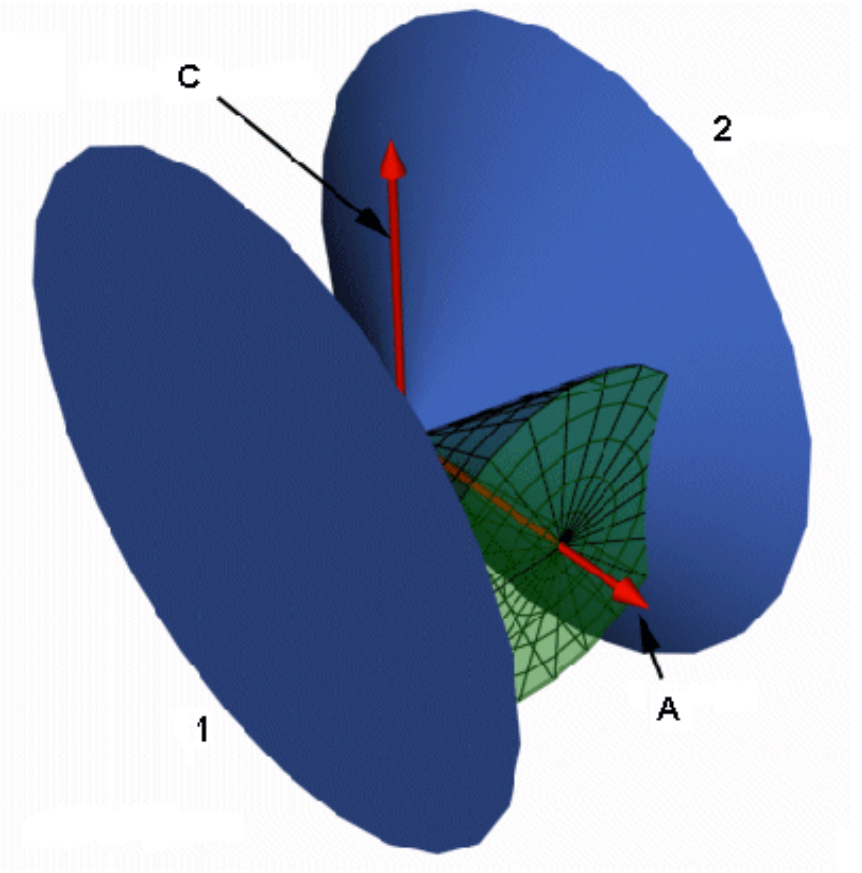
C. Twist x Plane Axis (outward)

1. Plane

2. Plane Min. Cone

3. Plane Max. Cone

This allows you to create two different scenarios; one where the cones generated by the plane limits intersect with cone produced by the cone angles and the one where they do not:

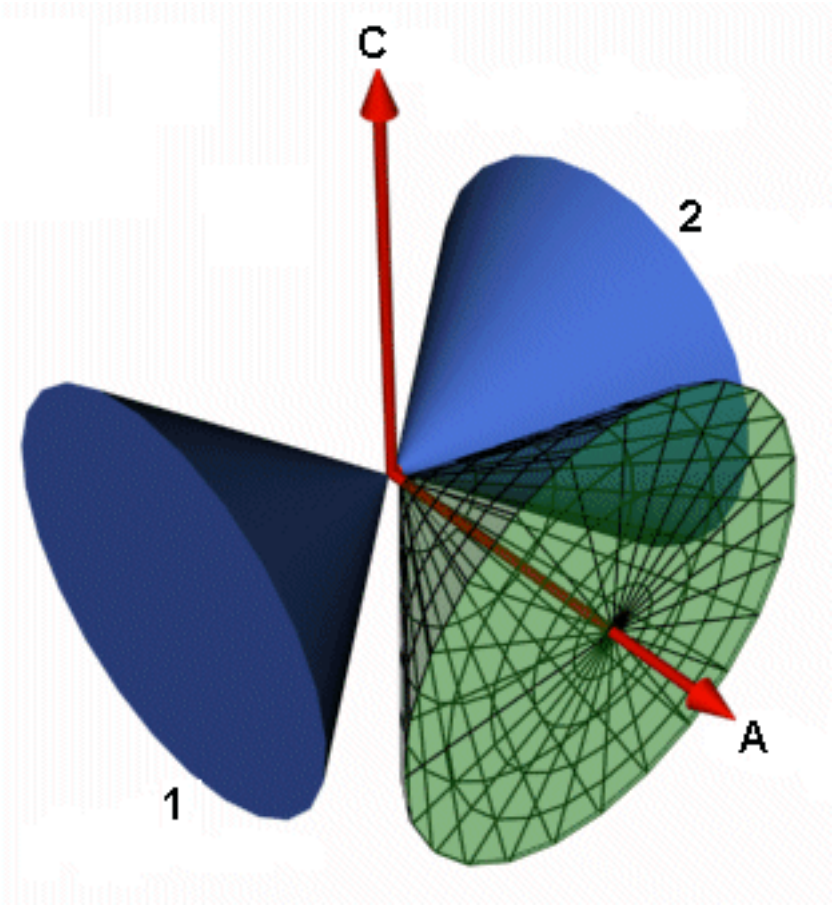


A. Twist Axis

C. Twist x Plane Axis

1. Plane Min. Cone

2. Plane Max. Cone



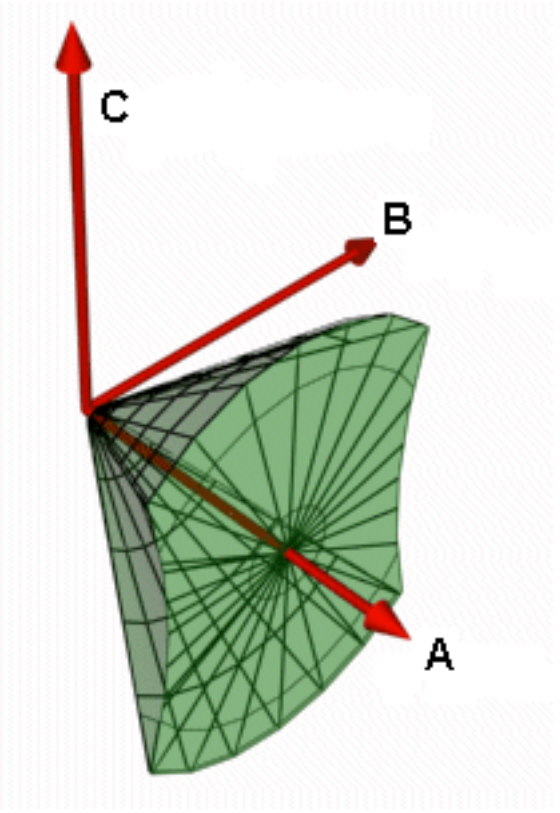
A. Twist Axis

C. Twist x Plane Axis

1. Plane Min. Cone

2. Plane Max. Cone

If the volumes do not intersect, the plane limits do not have any effect in the constraint; however, by allowing the cones to intersect, and deeming the overlapping volume as invalid body positions, you can limit the allowed positions for the child body further still.



A. Twist Axis

B. Plane Axis

C. Twist x Plane Axis

These are the parameters that you will be adjusting in reactor to tweak the behavior of the character. Hopefully this brief overview will help clarify Rag Doll constraints a little for you.

Setting Up Hips, Back, Neck and Shoulders

Open the sample file:

- You can either continue from where you left off or load *character02.max* from the *\dynamics_and_effects\reactor\ragdoll* directory.

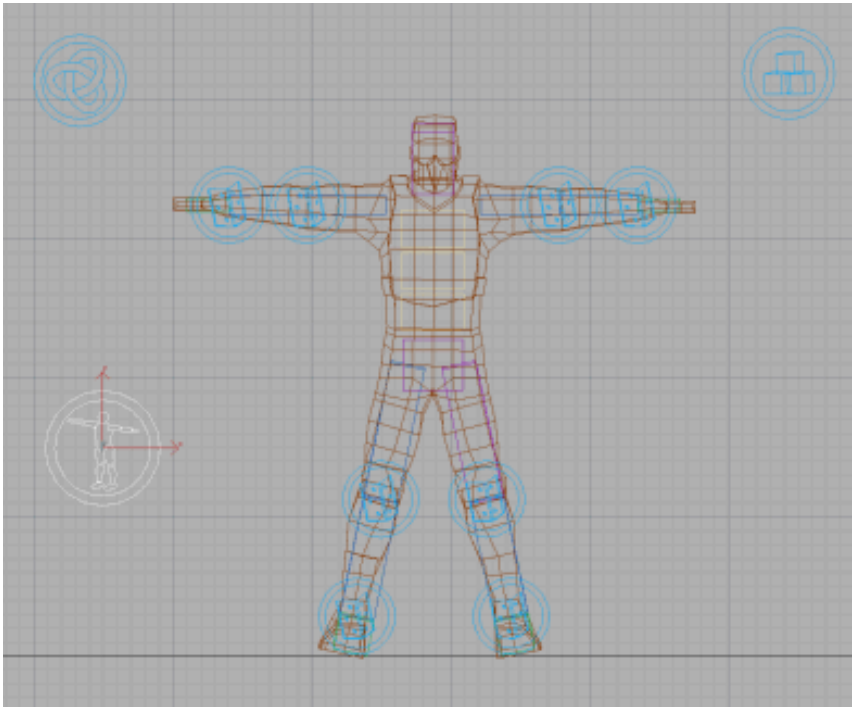
You will approach this lesson in a similar manner to the hinge constraint lesson in that you'll go through one of the joints in detail and then use a table of parameters to construct the other constraints. The setup for each Rag Doll constraint is identical, so once you've mastered the first one you should have no trouble setting up the rest of the character. You start with the hips.

Create Rag Doll constraints for the hips:

1  On the Create panel, click Helpers and choose reactor from the drop-down list.

2 On the Create panel > Object Type rollout, click Ragdoll.

3 Click in a viewport to create a Ragdoll helper object.



Next, you choose which objects to associate with this constraint.

4 On the Modify panel > Properties rollout, click the button labeled None, next to Child.

5 Press H, choose **RThigh** from the Pick Object dialog, then click Pick.

The button beside Child should now read **RThigh**.

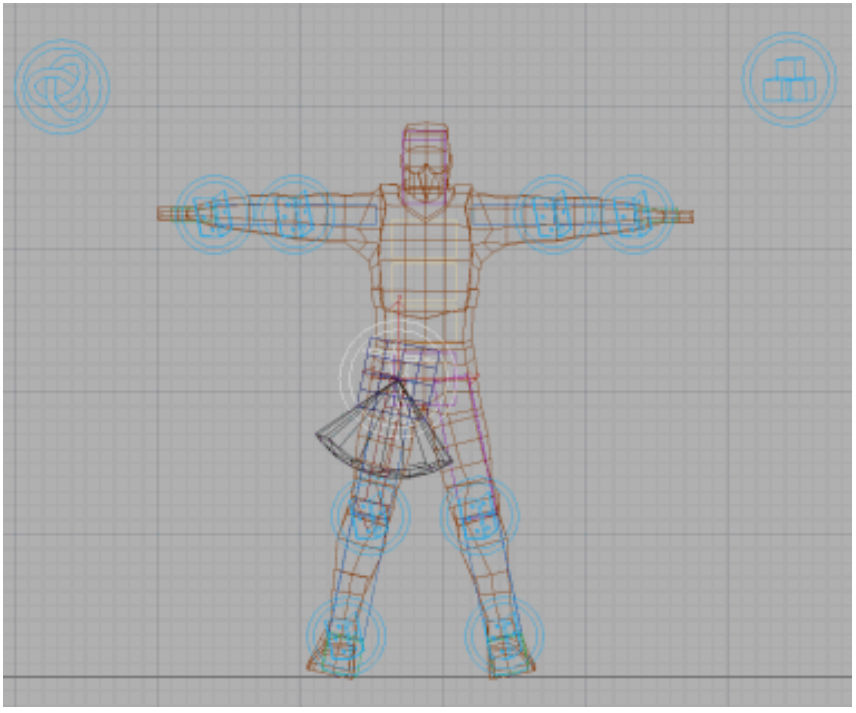
6 Turn on Parent.

This tells reactor to constrain two bodies together.

7 Click the button labeled None, next to Parent.

8 Press H, choose **Pelvis** from the Pick Object dialog, then click Pick.

The button beside Parent should now read **Pelvis** and the scene should look like this:



Next, you need to adjust the Rag Doll constraint limits so that it behaves like a real hip joint:

9 In the Modify panel > Properties rollout > Limits group, set Twist Min to **–5.0** and Twist Max to **5.0**.

10 Set Cone Min to **–25.0** and Cone Max to **55.0**.

11 Set Plane Min to **–45.0** and Plane Max to **2.5**.

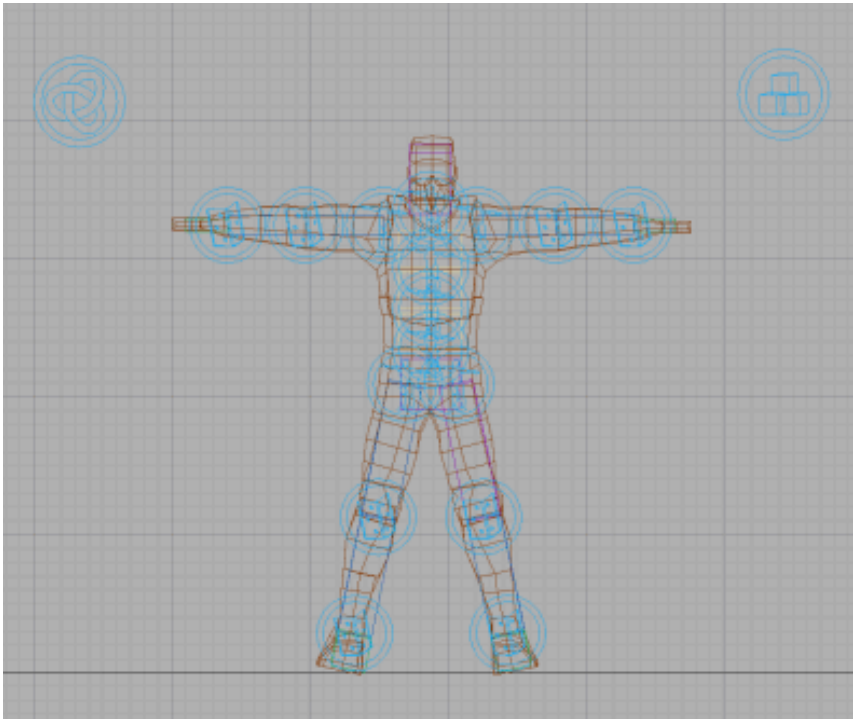
You have completed your first Rag Doll constraint. Once you have successfully finished with that constraint, you can move on to completing the rest in a similar fashion.

Create Ragdoll constraints for the back, neck, and shoulders:

- The table below gives a list of the constrained bodies, the constraint name and the constraint parameters. Use this table to set up the remaining Rag Doll constraints in your scene.

Constraint Name	Child	Parent	Twist Min	Twist Max	Cone Min	Cone Max	Plane Min	Plane Max
RHipJoint	RThigh	Pelvis	–5.0	5.0	–25.0	55.0	–45.0	2.5
LHipJoint	LThigh	Pelvis	–5.0	5.0	–55.0	25.0	–45.0	2.5
BackJoint1	Spine1	Pelvis	–1.0	1.0	–30.0	30.0	–50.0	5.0
BackJoint2	Spine2	Spine1	–1.0	1.0	–8.0	8.0	–6.0	3.0
BackJoint3	Spine3	Spine2	–1.0	1.0	–8.0	8.0	–6.0	3.0
RShoulder-Joint	RUpper-Arm	Spine3	–5.0	5.0	–75.0	85.0	–5.0	65.0
LShoulder-Joint	LUpper-Arm	Spine3	–5.0	5.0	–85.0	75.0	–5.0	65.0
NeckJoint	Head	Spine3	–5.0	5.0	–45.0	45.0	–15.0	15.0

Once you've created all of the above constraints your Rag Doll constraints should look like this:



The only thing left to do is to add these new Rag Doll constraints to your Constraint Solver.

Add the Rag Doll constraints to the Constraint Solver:

1. Press H and select the **CSolver01** object.
2.  On the Modify panel > Properties rollout, click Add.
3. Select all the objects (Rag Doll constraints) and click Select.

   Now the Constraint Solver has all of the constraints in it and the scene is ready to preview once again. First, save your scene.
4. Save the file as **my_character03.max**

Preview your character:

1.  On the Utilities panel > reactor > Preview & Animation rollout, click Preview In Window.
2. Press P to view the animation.

   Now your character should tumble down the stairs properly.

Creating the Animation

Now that the simulation is working correctly, it's time to create the animation. Since you don't want to see the bone objects and the constraints in your animation, the first thing you'll do is hide them.

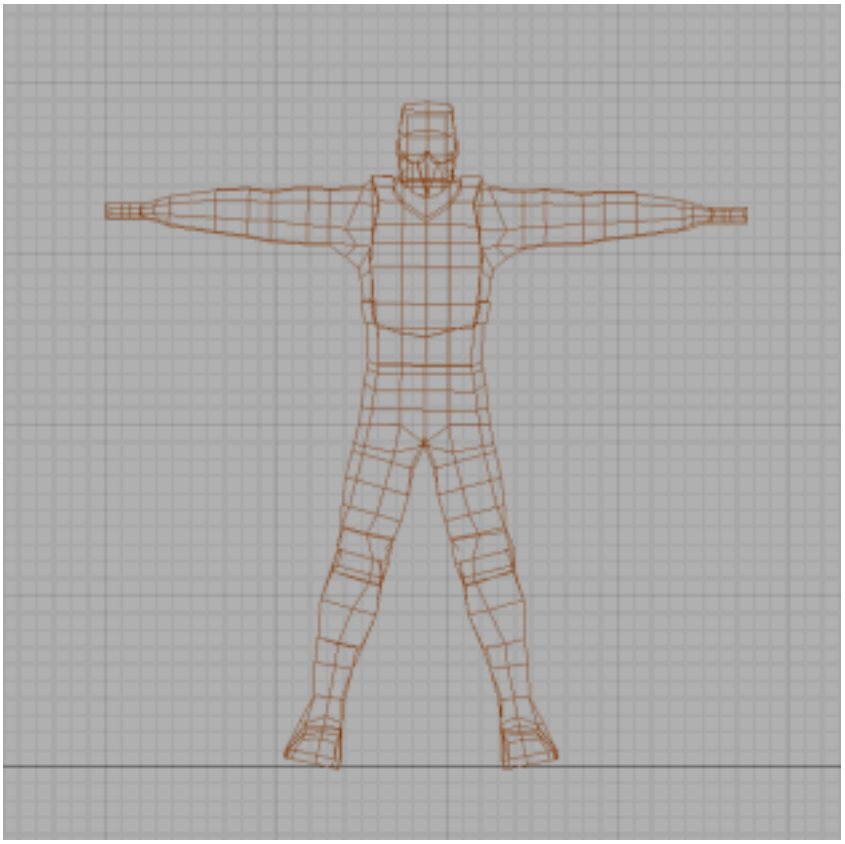
Open the sample file:

- Continue from the previous lesson or open *character03.max* from the *\dynamics_and_effects\reactor\ragdoll* directory.

Hide the bones of the character:

1. Press H, choose the **skin**, the **floor**, and all 16 **steps** on the Select From Scene dialog, then click OK.
2.  On the Display panel > Hide rollout, click Hide Unselected.

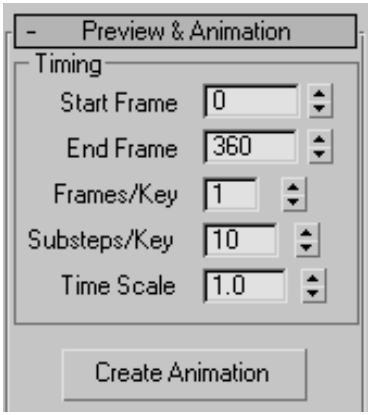
   Now you should only be able to see the skin of your rag doll.



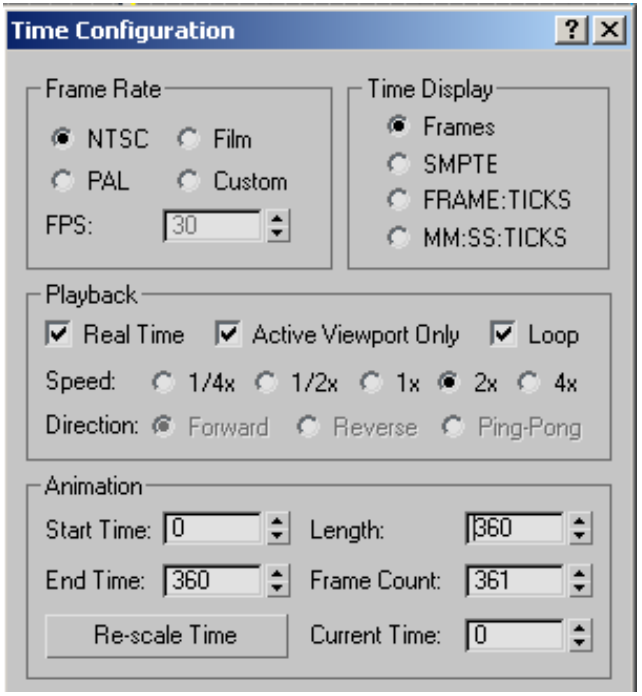
Create and play the animation:

The animation comprises roughly 360 frames, so before we play back the animation in a 3ds Max viewport, we'll need to adjust the animation length. We'll also adjust the playback speed for added realism.

1  On the Utilities panel > reactor > Preview & Animation rollout > Timing, set End Frame to **360**.



2 Click the Time Configuration button and in the time Configuration dialog > PLayback group, set Speed to **2x** and in the Animation group, set Length to **360**.



3  On the Utilities panel > reactor > Preview & Animation rollout, click Create Animation.

4  Once the animation calculations have completed, click on the viewport of your choice and click Play.

And that's a wrap! It's up to you now to tweak the various parameters on the rag doll to alter its behavior as it tumbles down the stairs. You may find some joints are too loose or tight: reduce or increase the limits accordingly. You can give both halves of the rag doll's body different settings to encourage it to fall in a given direction. Try changing its initial position and orientation or add some obstacles on the stairs.

You can see the final result of this tutorial by opening *characterEnd.max* from the *\dynamics_and_effects\reactor\ragdoll* directory.

Summary

You can use the reactor constraints to hold a character together, and control how it behaves during falls, collisions, and other physical encounters.