Chapter 10 – Physics

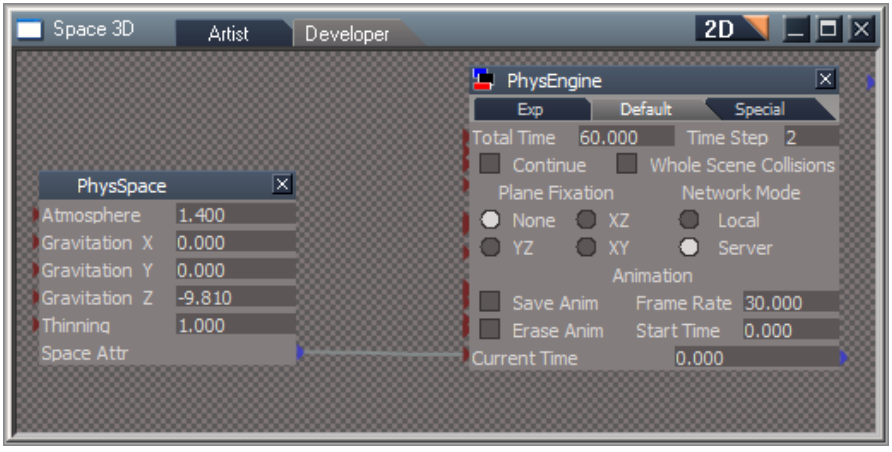
10.1.1 Physics: Space and Engine

Physical Simulation in trueSpace allows you to simulate the realistic behavior of objects according to the laws of physics. You can create real-time realistic movement of objects in the Workspace, taking into account the presence of gravity and the forces originating from the interactions between objects (collisions). You can directly interact with those objects during the simulation as well.
Physics in trueSpace comprises various attributes associated with the handling of objects and environments. Before we move ahead with the interesting aspects of physics, we should cover these important attributes and toolsets, to give a good foundation for exploration and experimentation with trueSpace physics.
At its most basic level, physics requires an engine and a space to work with; it also requires objects that have physical properties set on them using the toolsets.

The **PhysEngine** object and the **PhysSpace** object define the basic physical parameters of the space in which the simulation will run. These objects are added automatically upon clicking the Physical Simulation icon. These parameters include the strength and direction of gravitational force, density of the atmosphere, and the thinning of the atmosphere with height (altitude).

The PhysEngine object in Space3D manages the physical simulation. This object is added automatically along with the **PhysSpace** object to a scene as soon as you click on the Physical Simulation icon, you do not need to add them yourself.

Start/Stop Simulation



Two objects that control Physical simulation

All objects in a scene are included in the simulation. Objects are divided into two categories: *active* physical objects and *passive* physical objects. Active objects have simple physical properties and the simulation process computes their position and rotation. Static objects cannot be moved (or rotated) by the simulation, but they can interact (collide) with objects that have physical attributes.

 **Add Phys Attrib**:

To make an object *active*, you must first assign a set of physical attributes to the object. These properties are **Mass, Elasticity, Friction, Coarseness**, and, **Resistance.** For each *active* object, you can also set some initial kinematic parameters, such as initial speed and acceleration and initial rotation and rotation acceleration, which will be involved in the physics computation at the start of simulation process. A special object, **PhysObj**, represents all of the attributes and initial parameters.

The Tools and Objects are covered in Detail later on in this section:

There is also a series of scenes available as a resource to use so that you can follow along with them and use as additional reference aids to this section of the manual.

You can unzip the files to the trueSpace76\tS\Rs Main Libraries folder and the library will show as a choice in the library browser called tS76Physics101 and this can be loaded in the library stack.
Alternatively you can unzip to a folder or library place of your own choice and then either open them through the library browser by adding a library place or import them to a new created library.
Or you can just drag and drop them from windows explorer folder you unzipped them to into the Workspace window .
The easiest method is placing the tS76Physics101 folder from the zip into the Rs Main Libraries folder, but the choices and alternatives are there as well if you want to use them.

- Reference : Libraries

 Scenes Resource Link

10.1.2 Simulation Controls : Engine and Space

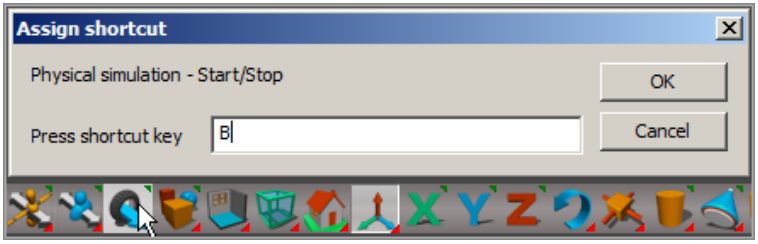
-  **Start/Stop Simulation , Create Physics Engine**

The Start Physics tool is a toggle for the engine. Two things happen when you click the Start Physics tool.
First if no engine exists the PhysSpace object and the PhysEngine object is created inside the Link Editor. These two objects are connected automatically and form the foundation for physics simulations in trueSpace.

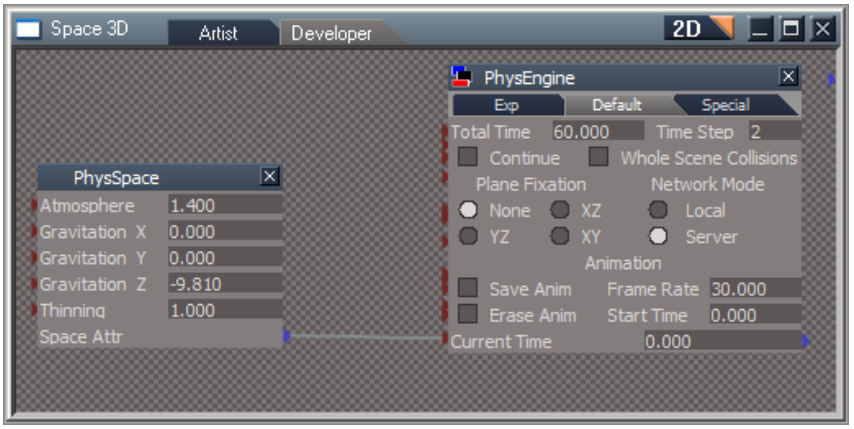
The simulation is run and stopped by using the toggle only after the Engine and the Space has been created.
The simulation will automatically stop after the time interval specified by the value of the Total Time attribute of the PhysEngine object. When you want to stop or interrupt a simulation manually before it is finished, click on the icon again. After stopping a physical simulation, all objects go to back to their initial position unless the Continue attribute has been checked. In this case, the objects stay in the position at which the simulation is stopped.

Tip: You can assign shortcut keys to Tools by holding down CTRL and R-Clicking on a Tools icon.

- Reference: Custom Shortcuts

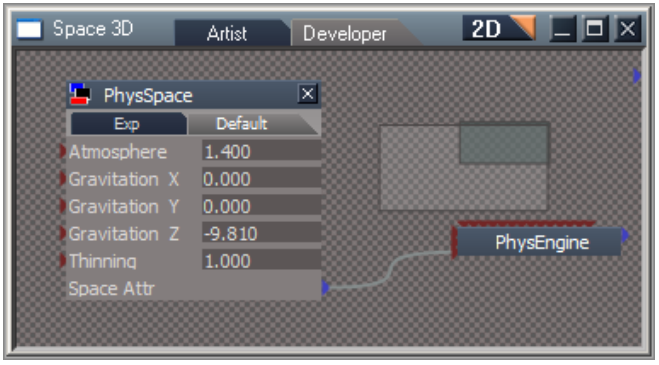


CTRL and R-Click to assign shortcut keys



Basic elements of Physics in trueSpace: PhysSpace object and PhysEngine object in default aspect

- **PhysSpace object attributes**

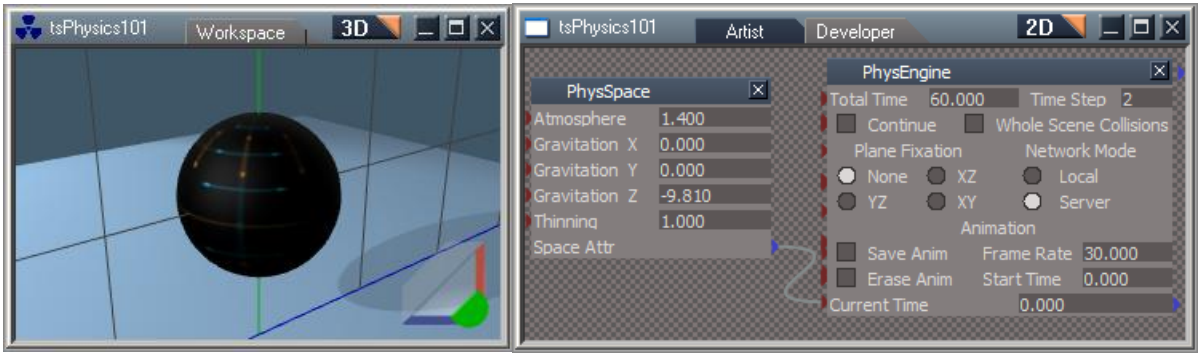


PhysSpace object

PhysSpace object, has the following attributes:

- **Atmosphere:** by default, atmosphere for trueSpace physics is set to a value of 1.400 which is trueSpace equivalent to Earth's 14.7 psi (pounds per square inch) or 101.3 kilopascals. Since atmosphere equals pressure, we are given the capability to change this value. To gain equivalent of space vacuum, enter a value of 0.000.
- **Gravitation:** the phenomenon where all objects are attracted to each other. In trueSpace you again control this phenomenon through the setting of values for 3 axis directions. By default, trueSpace approximates Earth's own gravimetric forces using the default values for these settings.
  - **X:** by default, value of 0.
  - **Y:** by default, value of 0.
  - **Z:** by default, value of -9.81. Important value for zero-gravity scenario. Set to 0.000.

- **Thinning:** it is possible to have more than one atmosphere at work in trueSpace. Individual objects may have a Local Environment applied to them. The Local Environment is paramount of a mini-atmosphere. How these atmospheres affect objects in close proximity is governed by Thinning setting. By default, trueSpace uses a value for the PhysSpace object of 1.000. Default for a Local Environment will be 0.000. Thinning works on "height" of object/local environment. Like Earth's atmosphere thins as you go higher, so can you approximate this using the thinning setting.
- **Space Attr:** all the above mentioned settings are packaged up into an ouput connector and passed along to the PhysEngine for processing. There is no way for you to adjust this setting except by adjusting the other settings.

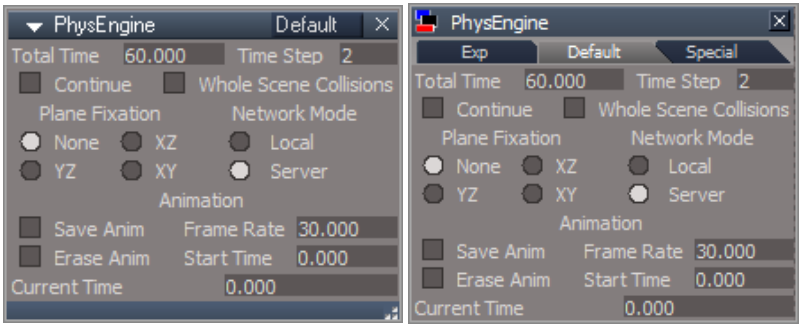


tsPhysics1a tutorial: explore PhysSpace

- video link

PhysEngine object attributes

The PhysEngine object manages the simulation and provides you with control over aspects and values the engine will use for processing the objects with physical properties assigned to them and physics in the space/scene during a Physical Simulation. PhysEngine is added automatically to a scene as soon as you click on the Physical Simulation icon. The engine contains three groups of controls sorted into Aspects/tabs for Exp, Default, Special.



PhysEngine object, Default aspect tab as seen in the Stack and the LE

PhysEngine Aspects:

- **Default Aspect :** holds the most commonly used settings:

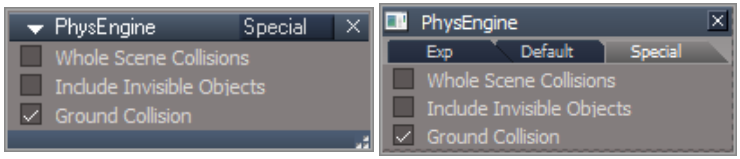
- **Total Time:** allows you to set total time for the simulation to run before it stops itself. By default 60 seconds is used. You can feel free to adjust this value to suit your needs
- **Time Step:** Value specifies how often the new position of any objects in the simulation will be computed per one frame; for example, a value of 1 means that one computation is performed for one frame, a value of 4 means that 4 computations are performed for one frame.
- **Continue:** use this setting to control whether a simulation starts or continues each time the simulation is started/stopped.
- **Whole Scene Collisions:** specifies whether collisions with all objects in the scene (including lights and cameras) will be considered during the simulation. If you turn off this attribute, all collisions between active objects and lights and cameras will be ignored.
- **Plane Fixation:** specifies whether physical simulation will run in 3D space or only in one plane (2D physics). In the Default Tab and default value, the attribute is set to "None" which means that the simulation will run in 3D space without any fixation to a plane. To set a fixation to one plane or 2D physics, set the value to xz, yz, or xy plane.it is possible to create scenarios that approximate real life, such as with the AirHockey game. In theory, a puck slides along the ice over a cushion of air. Using Plane Fixation accomplishes this in the AirHockey game, which is located in the Scenes – Active library. On the Default aspect of the engine, you have 4 choices for Plane Fixation:
    - **None:** value = 0.
    - **XZ:** value = 1 and physics "action" is restricted to the XZ plane.
    - **YZ:** value = 1 and physics is restricted to the YZ plane.
    - **XY:** value = 3 and physics is restricted to the XY plane. The AirHockey scene uses this setting.
- reference: Ch_11_5_3_AirHockey
- **Network Mode:** This attribute is only used when trueSpace client is connected to a shared space the values set here are ignored when the physics engine is run locally on a machine.
- When connected to a shared space the settings used are
    - Local - the simulation runs only on the client . The other clients can see the simulation but they can't start and stop the physics or interact using the phys move tool. (value 0, in the Exp aspect).
    - Server - the simulation runs on the server. The other connected clients can start and stop the simulation and also interact with the physics objects using the phys move tool. . (value 2, in the Exp aspect).

- **Save Anim:** when checked, this setting triggers keyframe creation in the AE. You can save the movements calculated by the physics engine to keyframes for later playback, keyframes for each active physical object will be generated and saved after the simulation is started in the Workspace. The saved animation sequence can then be later replayed in the Anim View , and can be used in the same way as any other keyframed

animations.

- **Erase Anim:** erase the previous keyframes from the AE (Animation Editor) if Save Anim is used.
- **Frame Number:** the physics engine has its own internal timer. Default value of -1 is used as a start point for calculating frames. It is possible to connect another timer object to this connector and feed it frame numbers, in which case the internal timer is disabled.
- **Frame Rate:** equals frames per second or **Frame Time** from the Expanded aspect. They are the same.

○ **Special Aspect :** Holds some additional settings for invisible objects and ground collision, Whole scene can also be found on the Default aspect

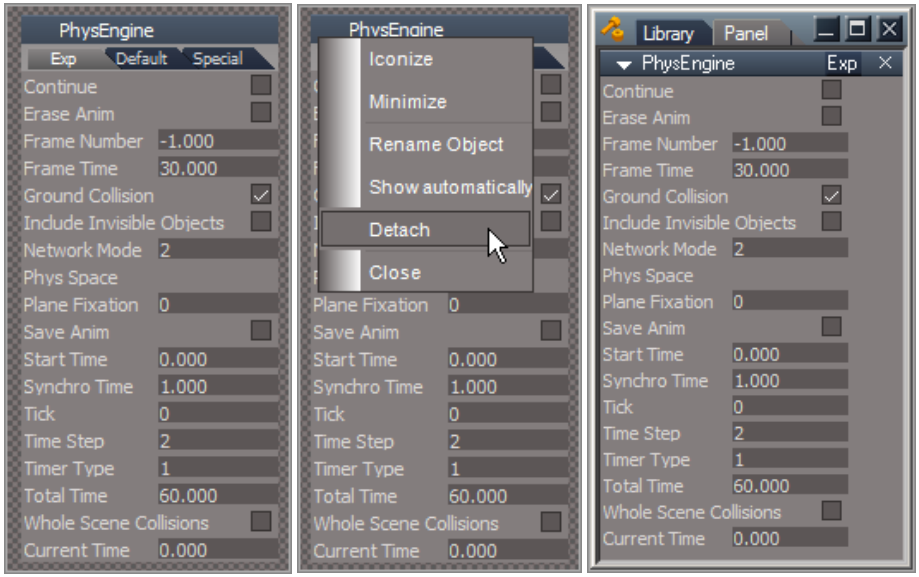


PhysEngine object, Special aspect tab as seen in the Stack and the LE

- **Whole Scene Collisions:** specifies whether collisions with all objects in the scene (including lights and cameras) will be considered during the simulation. If you turn off this attribute, all collisions between active objects and lights and cameras will be ignored.
- **Ground Collision:** Enable or disable collision with the Ground.
- **Include Invisible Objects:** Choose whether invisible objects participate in the physics collisions.

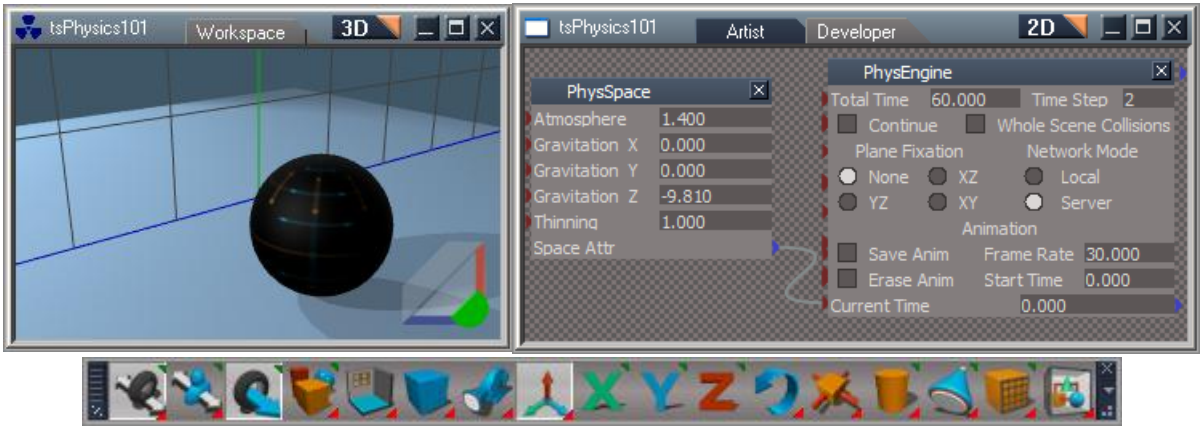
○ **Expanded Aspect :** Contains all of the settings from the other two panels and it also has some additional attributes. Most of the time you do not need to use the Expanded aspect as the more common and most used controls are found on the Default and Special Aspect tabs.

Note: To save repetition see the Default and Special aspect for descriptions of the setting.
Note and Tip:- This Aspect doesn't show as available in the stack unless you choose to detach it from the menu in its title bar



PhysEngine object, Exp aspect tab as seen in the LE and the Stack if detached is used

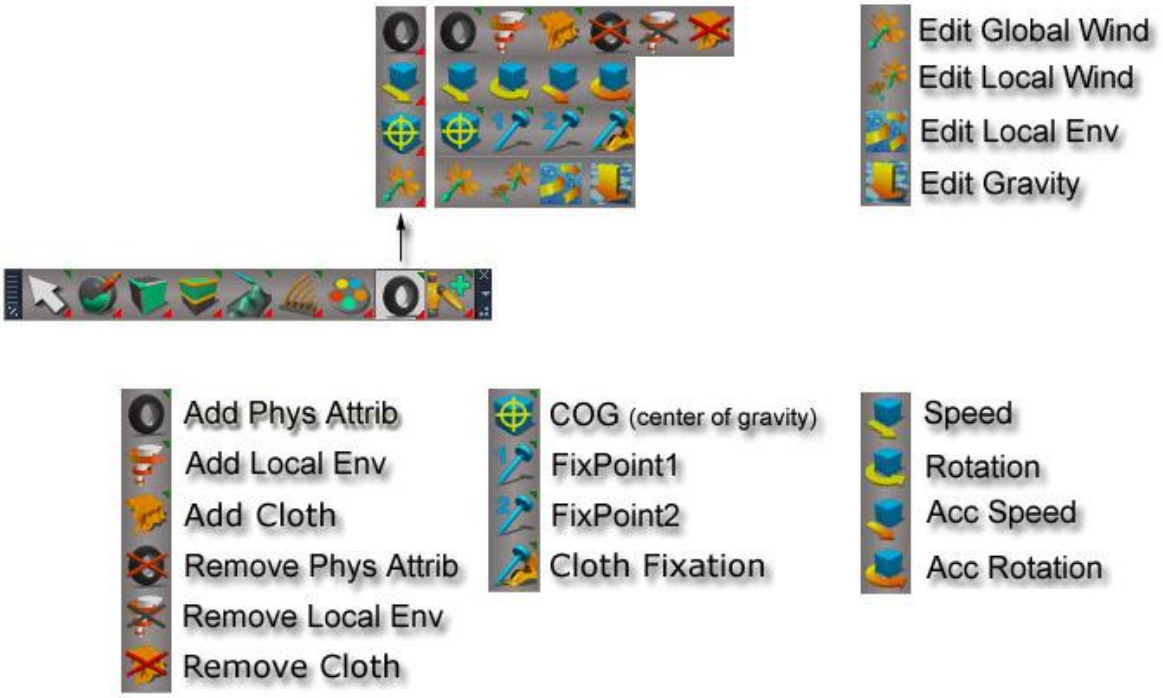
- **Frame Time:** equals frames per second or **Frame Rate** from the Default aspect. They are the same.
- **Network Mode:** There are two viable values that you are not so much concerned about here. Value of 0 would indicate Local and a value of 2 would indicate Server is checked from the Default aspect of this object.
- **Phys Space:** this is connector that feeds off information sent from the PhysSpace object in Link Editor. Nothing to change or set here, just info used by the engine in calculating simulation within the space.
- **Start Time:** used by physics in complex scenario when server is being used. Not something you would set arbitrarily. trueSpace handles this one for us.
- **Synchro Time:** once again a setting used by trueSpace when server is involved. Let trueSpace take care of this setting.
- **Tick:** used by server scenario and not a setting you would be adjusting. Server must co-ordinate physics for multiple possible users in a Shared Space scenario. This setting is best left for trueSpace to manage.
- **Timer Type:** signifies if internal timer is being used or if a more complex server scenario is being used in the scenario to feed frame values to the engine. Once again trueSpace will manage this for you but it is available for viewing.
- **Current Time:** this output connector can be used by scripts or such as a value for time. This can help you sync other elements of your scene to work with physics.



tsPhysics1b tutorial: explore PhysEngine

- **video link**

10.2 Physics Toolbar



Main Physics Tools

Physics Move and Start/Stop simulation tools

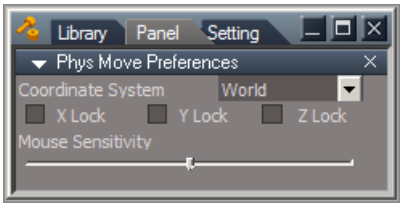
Create Physics Engine: Start /Stop Simulation :

L-Click creates the physics engine and starts and stops the simulation. R-Click shows a Panel in the stack for setting values for how the engine behaves during the simulation.

Physics Move :

L-Click to activate, allows for objects with physical properties to be interacted with during a simulation by "grabbing " them with the mouse in the Workspace window. This tool can only be used only when a simulation has

started and the physics engine is at work. R-Click on the icon shows a Panel in the stack for setting preferences to how the tool behaves. You can choose to use World or Screen coordinates and choose to lock individual axis to restrict the mouse interaction , you can also adjust the mouse sensitivity. The basic settings are enough for most purposes.



Physics move tool preferences

Using the Physics move tool with World coordinates: L-Button held and dragged allows the object to be freely moved or according to locks restrictions , R-button and drag allows for movement restricted to Z.
Using the Physics move tool with Screen coordinates: L or R Button held and dragged allows the object to be freely moved or according to locks restrictions.

Note: Recording physics keyframes will include any interaction between the user and the objects using the Phys Move Tool, but will NOT include any movement of objects using the Object Move tool. This means that only the behaviors of Active Objects are recorded.

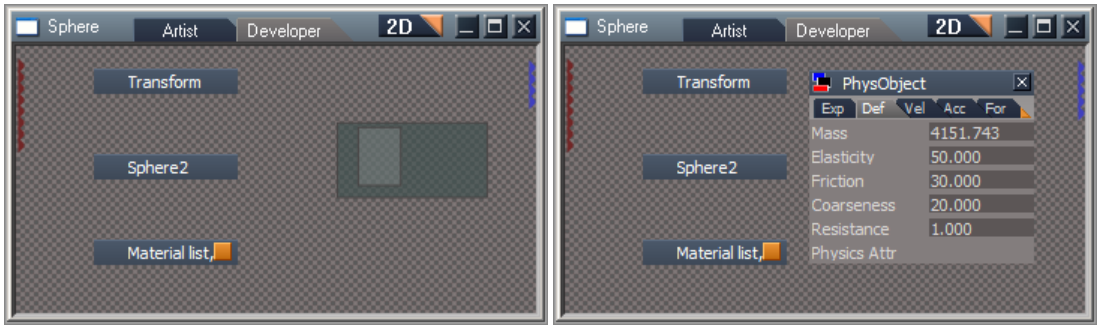
As well as the Start and stop , and the Phys Move icon there are four groups of tools for setting initial parameters for objects which have physical properties assigned to them.

10.2.1 Solid Objects

 **Add and Remove Physics Objects**

 **Add Phys Attrib**:

L-clicking on Add Phys Attrib tool adds the default solid structure for Physics Attributes to the selected object , and the PhysObject is created inside the object in the LE. When you use the Add Phys Attr tool, this object will always be titled PhysObject . Values for various physics attributes are calculated based on the object's size/mass.
Values can be edited either in the panel in the stack or in the Link Editor.

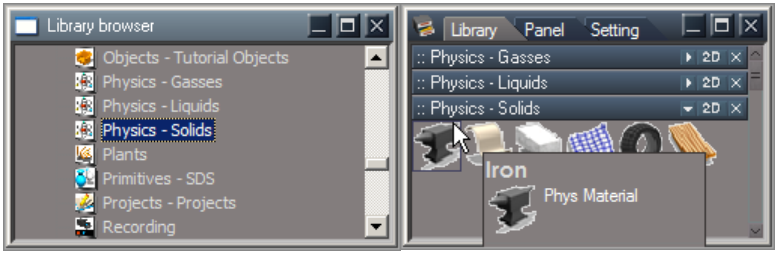


PhysObject is created inside the object in the LE, a Sphere in this case

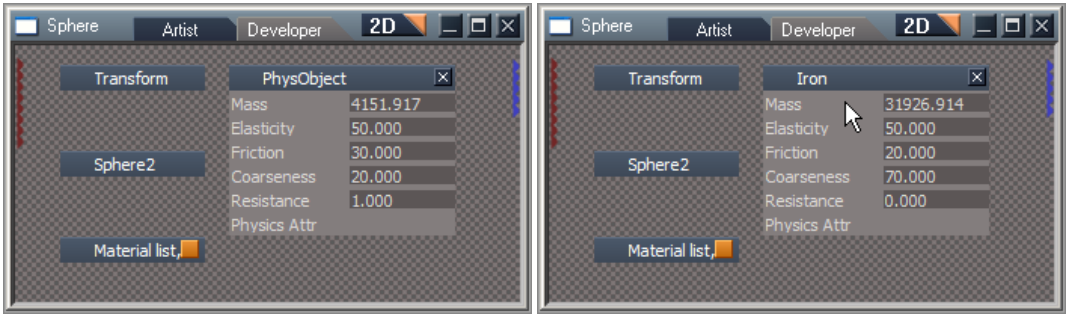
Turning objects into Solids:

To use the term "Solid" reflects how the object will react and be used in physics: solid as opposed to a liquid or a gas. To assign physical attributes to an object in the Workspace, first select the object and then click on the Physical Attributes tool icon.
Physical properties can also be added to a static object during a simulation by selecting it and pressing the Physical Attributes tool icon. The static object changes its state to active, and it is then included immediately in the simulation. Another way to turn objects into solids is to use the Physics Materials library.

There are six materials in this library: **Rubber**, **Iron**, **Polystyrene**, **Paper, Rag** and **Wood**. Materials differ by settings for Elasticity, Resistance, Friction and Mass. You can simply drag and drop these materials onto objects in the Workspace window, and ordinary objects turn into physic solids with the relevant parameters pre-applied.
If your object already has a default Physics solid applied the name of the PhysObject will change according to the material used, below its been replaced by the iron object. Essentially the Physics materials all have the same attributes available but with differing default values for the material being used, you can use these as a starting point when experimenting. Take a moment to examine the settings used for each material.



Physics - Solids Library

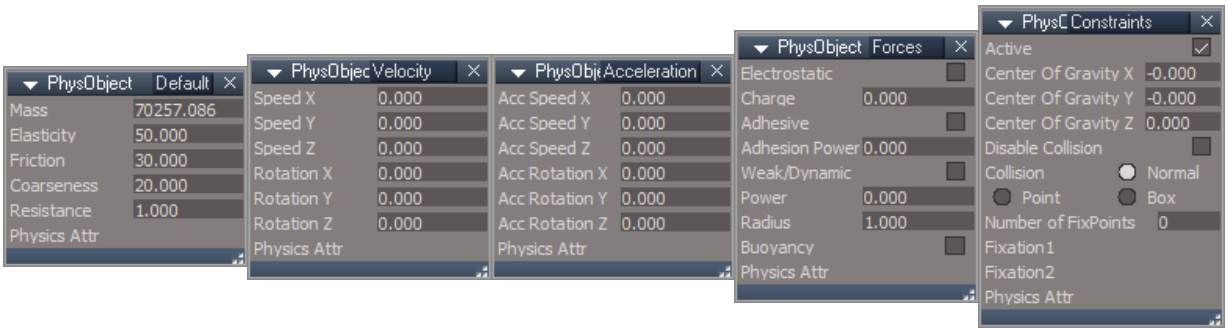


Apply iron to the Sphere and the PhysObject title changes to Iron

- Reference: Tutorial Physics Solids
- Reference: Physics Materials Libraries

By default, the PhysObject object displays the Default aspect. There are 6 aspects to this particular object. Five of them can be seen in the image below, with the sixth aspect (Constraints) accessible via the small orange triangle button/area to the right of the For (Forces) aspect.

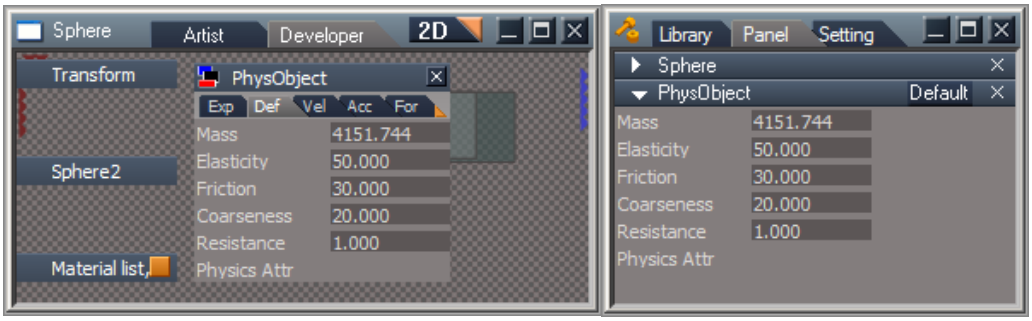
The Expanded aspect is a collection of all the settings from all the attributes. It is rather long in size in the Link Editor. To better present these settings, the other five aspects separate out the various settings into categories or groups that have a common function or behavior.



PhysObject has 5 specific tabs with attributes unique to that tab

- **PhysObject attributes and tabs:**

- Default



PhysObject Default aspect in the LE and the Stack

- **Mass:** every object with physical attributes must have a mass of some value. By default, trueSpace will assign a value for Mass using a calculation based on the size of the object.

- **Elasticity:** sets the value for how elastic the object is. **Elasticity** specifies how much energy of an object is preserved during collisions with the ground or other objects. If Elasticity is set to a value of 0, no energy is preserved and collisions are inelastic, i.e. after a collision, an object will simply stop, rather than "bounce off." Higher values of Elasticity mean that collisions will preserve more energy, all the way up to a value of 100.0, which means than no energy is lost during collisions. At this setting, an object will bounce off after a collision with as much energy as it had before the collision. If you set Elasticity higher than 100.0, the object's energy will actually increase after each collision, for example, a bouncing ball will keep getting higher and higher with each bounce! If this were a ball, how much would the ball bounce after being dropped on the floor? Higher settings would result in more bouncing, while lower settings would result in less of a bounce.

- **Friction:** sets the value for the friction element of physics. All objects have a friction factor of some value and you can control how a cube slides down a stick of lumber on a slant in a scene. As you reduce the friction and perhaps the angle the lumber is sitting at, the cube would begin to show signs that gravity forces are overtaking friction forces and the cube begins to head downward along the stick of lumber.

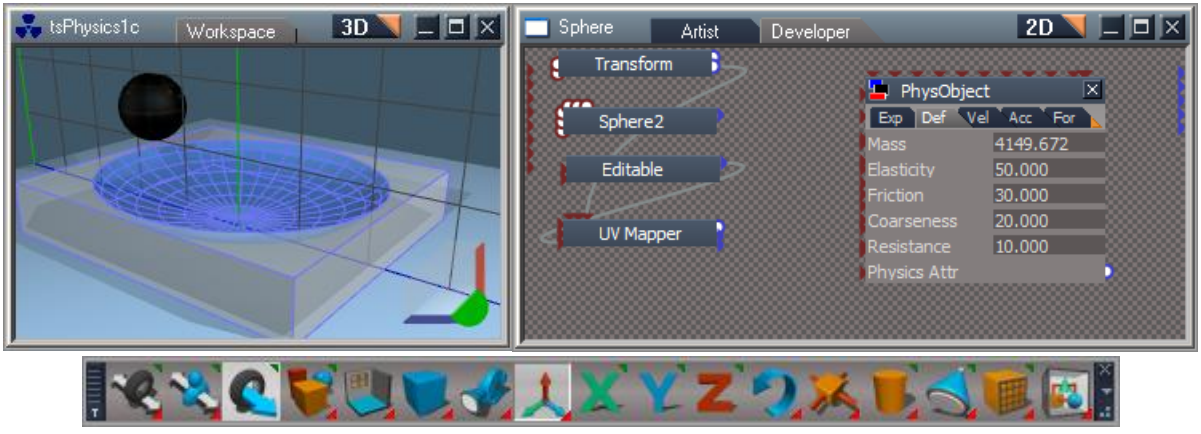
- **Coarseness:** works in conjunction with friction. It is easier to move a fine-grit piece of sandpaper across the surface of a piece of wood, that it is to move a coarse grit sandpaper across same piece of wood. The same theory applies to trueSpace friction. Two spheres set to collide with other in a "glancing blow", with different Coarseness values would cause the other sphere to react and spin after the collision in different ways. The coarser the sphere, the more influence it would have on a lesser coarse sphere.

  Note : **Friction** and **Coarseness** reflect the friction of objects in contact with other objects. The

Friction value is applied when objects are moving (called dynamic friction), and Coarseness if objects are in rest (called static friction). The values of Friction and Coarseness should be in the range from 0.0 - 100.0, with 0.0 meaning no friction, and 100.0 meaning maximum friction.
There is an interesting implementation of friction (dynamic friction applied for objects with nonzero speed) and coarseness (static friction for objects with zero speed) effects for active objects in physical simulation.
The main advantage of the friction model is a simpler relation between values of an object's friction attributes and behavior of objects in simulation – value of 0 means no friction for the object, whereas higher values effectively constrain the movement of objects along any ground or other objects it comes in contact with.

- **Resistance:** in real world physics, a *newton* is the amount of force required to accelerate a body with a mass of one kilogram at a rate of one meter per second squared. Resistance is greatest with a stationary object. How much force is needed to begin movement of the object. The Resistance value you apply to any object in trueSpace determines how much force will be required to move the object. Higher values are more difficult to move, while lower values make objects easier to move. This Value affects the resistance of objects during interaction with the atmosphere or local environments.

  Note: The value can be from range 0.0 – 1000.0. A Resistance of 0.0 means no resistance for a given object, and atmosphere and local environments will have no effect on the object. As the value increases, the resistance of object will be higher, and atmosphere and local environments will have an increasing effect on the object

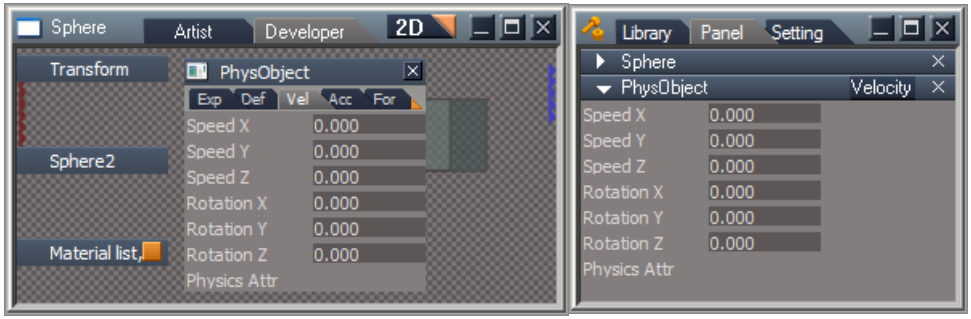


tsPhysics1c tutorial: explore PhysObject Default aspect

- video link

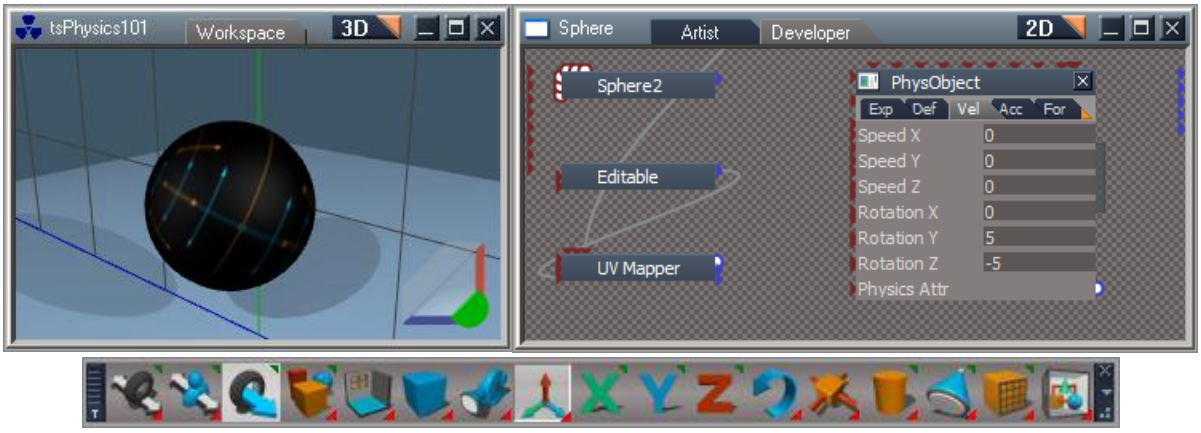
Velocity and Acceleration : Initial speed and acceleration of the object along the X, Y, and Z axes of the world coordinate system can be set by the attributes *Speed X, Speed Y, Speed Z, Acc Speed X, Acc Speed Y,* and *Acc Speed Z.* Similarly, initial rotation and rotation acceleration are set by the attributes *Rotation X, Rotation Y, Rotation Z* and *Acc Rotation X, Acc Rotation Y,* and *Acc Rotation Z.*

- Velocity



PhysObject Velocity aspect in the LE and the Stack

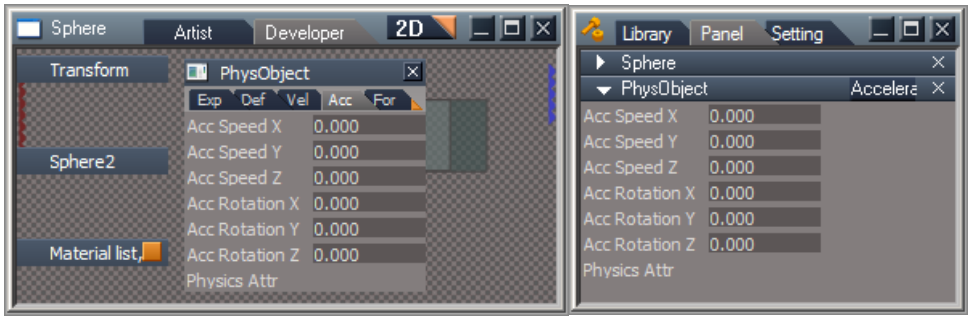
- **SpeedX:** initial speed in the direction of the x-axis for the object. Think of this as power in direction of X. When you combine different powers for XYZ, the object will move in only one direction, arrived at by computing power values for speed in XYZ.
- **SpeedY:** initial speed in the direction of the y-axis.
- **SpeedZ:** initial speed in the direction of the z-axis.
- **RotationX:** initial rotation motion in the direction of the x-axis. Just as with the values for speed, think of the rotation values as power to spin the object in direction of single or multiple axis directions.
- **RotationY:** initial rotation motion in direction of the y-axis.
- **RotationZ:** initial rotation motion in direction of the z-axis.



tsPhysics1d: explore PhysObject Velocity aspect

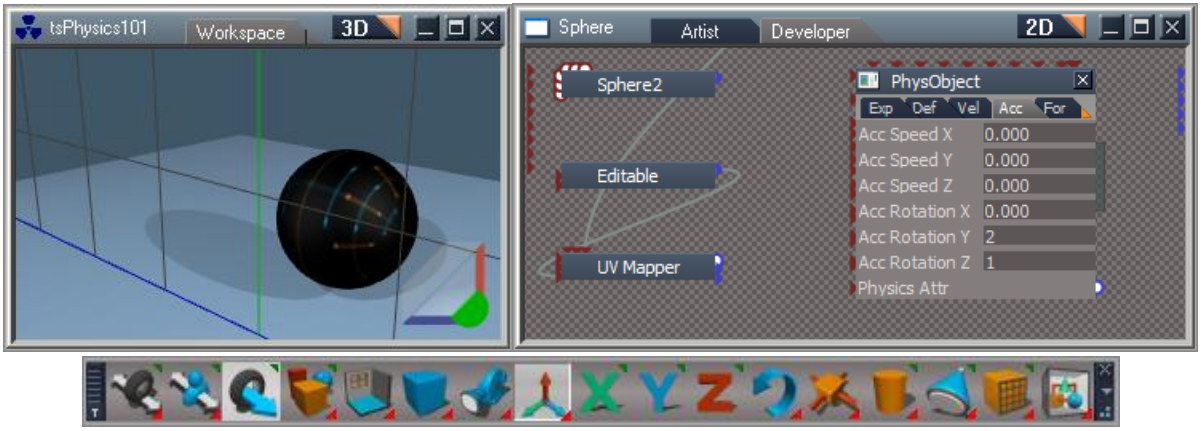
-  **video link**

- Acceleration



PhysObject Acceleration aspect in the LE and the Stack

- **Acc SpeedX:** set the acceleration for speed in x-axis direction. Designed to work in conjunction with the Speed settings, you have further control to add an acceleration to the direction. Just as with speed, you can vary the acceleration and use it in more than one axis. Once again trueSpace physics engine will do the math and decide which direction the object will move and how fast.
- **Acc SpeedY:** set the acceleration for the y-axis direction.
- **Acc SpeedZ:** set the acceleration for the z-axis direction.
- **Acc RotationX:** set the acceleration for rotation in the x-axis.
- **Acc RotationY:** set the acceleration for the rotation in the y-axis direction.
- **AccRotationZ:** set the acceleration for the rotation in the z-axis direction. Just as with the other values here, combinations can be used and trueSpace will calculate acceleration for all axis directions and send the object along accordingly.



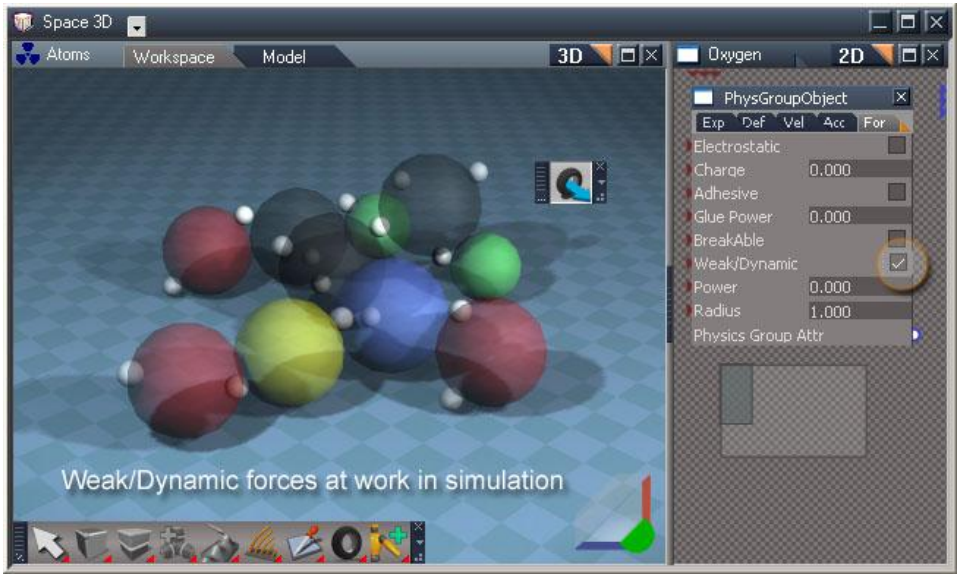
tsPhysics1e: explore PhysObject Acceleration aspect

- **video link**

- Forces

  There are three kinds of specific forces or effects that can act between objects during a simulation: Electrostatic, Adhesive, and string-like force called Weak/Dynamic. Once selected or "checked", their respective settings will take effect.

  To set Electrostatic effects for a given object with physical attributes, first check *"Electrostatic"* in the PhysObject . Then you can set the value of the electrostatic charge (positive or negative) for the *"Charge"*. During a simulation, objects with electrostatic charges will attract or repel each other. If one object has a positive charge and second a negative charge, they will attract each other. If both objects have negative or positive charges, they will repel each other. In simulations electrostatic forces act only between objects with *Electrostatic* checked.

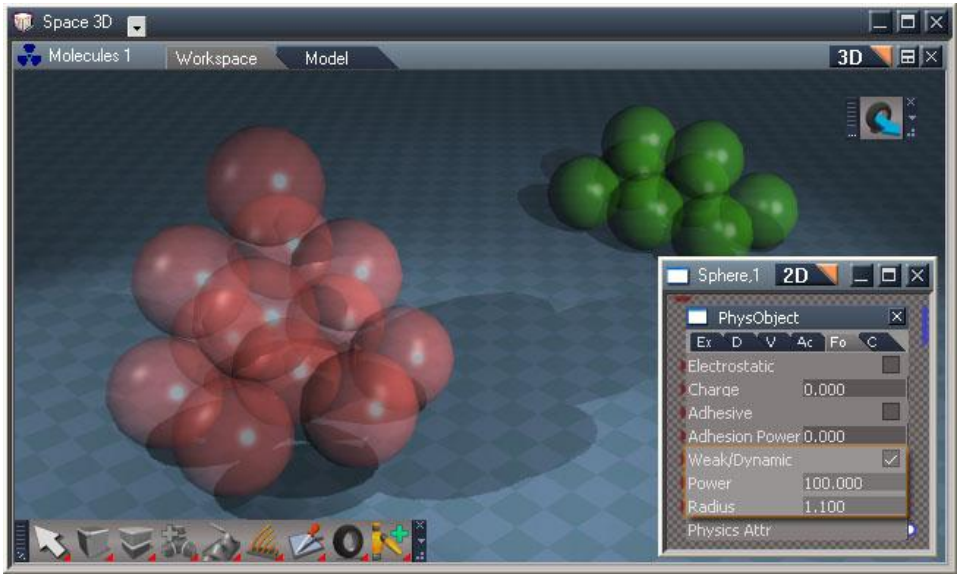


Time Lapse example of same Electrostatic Charges in action.

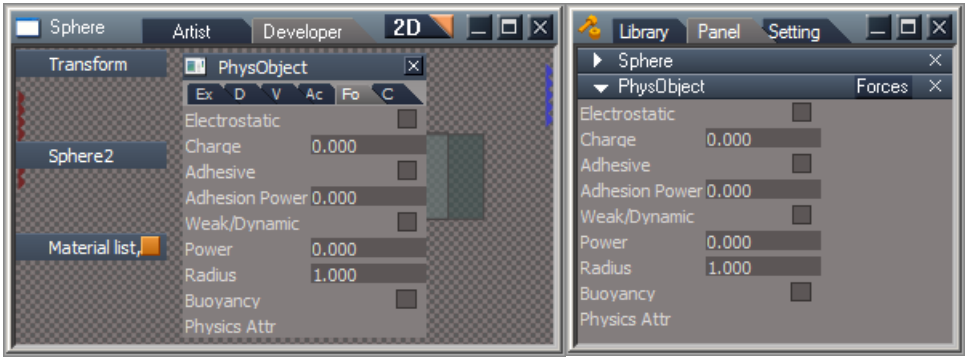
To set a weak force for an object with physical attributes, check the attribute *"Weak/Dynamic"*. Then you can specify values for the two input attributes: *Power* and *Radius*. *Power* sets the intensity of the force, while *Radius* sets a scope of force from the object (higher values mean that the force acts for a larger distance from the object). In simulation, string-like forces act only between objects with the *Weak/Dynamic* attribute checked.

For group physics objects, you can set different values of string power and string radius for each sub object. To do this, you should enter the sub-object's in LE and edit the relevant PhysObject object. See Sub-objects with physical attributes

The stability of a simulation with string effects depends strongly on the values of string power and string radius. If you wish to use strings for grouped objects, to increase stability of objects grouping you should decrease these values for given objects.
You can also increase the stability of a simulation with string effects by switching off gravity in the scene by doing any of the following:
Setting zero values in the input attributes of the PhysSpace object that is connected to the PhysEngine
Excluding collisions between objects. (To do this, check the input attributes "Disable Collision" of the PhysObject object for given objects.)
A good example of the string like effect of weak force is the Molecules1 scene in the Base scenes library (image below). You can simply enter this scene and activate the simulation. You should see that two groups of spheres adhere nicely together. It is also fun to drag individual molecules with the physics Move tool.



String-like Weak/Dynamic force is great for molecules

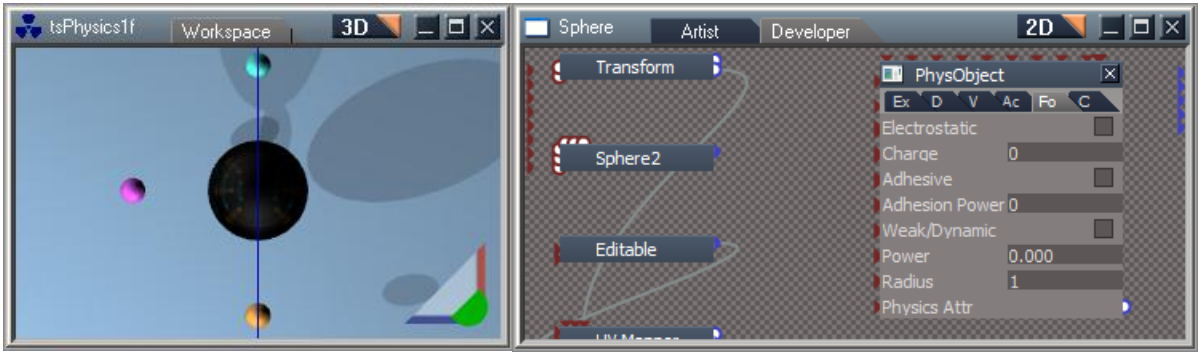


PhysObject Forces aspect in the LE and the Stack

- **ElectroStatic:** when you were young and you dragged your feet across the carpet, you built-up static electricity until you touched your siblings and discharged that static electricity in the form of a shock. The checkbox for ElectroStatic determines if this force will be present for this object in your physics simulation. When a charge is applied to objects with Electrostatic applied, every collision that occurs, reduces the value of Charge.
- **Charge:** just as with all power scenarios, there is a negative and positive charge at work. By default the value is neutral at 0.000. Values close to 0.000 provide very subtle attraction and repulsion, while values further away from 0.000 provide more powerful forces. Value for Charge remain constant until the Electrostatic check-box is selected, at which time the value for Charge is reduced with every collision.
- **Adhesive:** this checkbox determines if forces of adhesion will be present for this object in your

physics scenario.

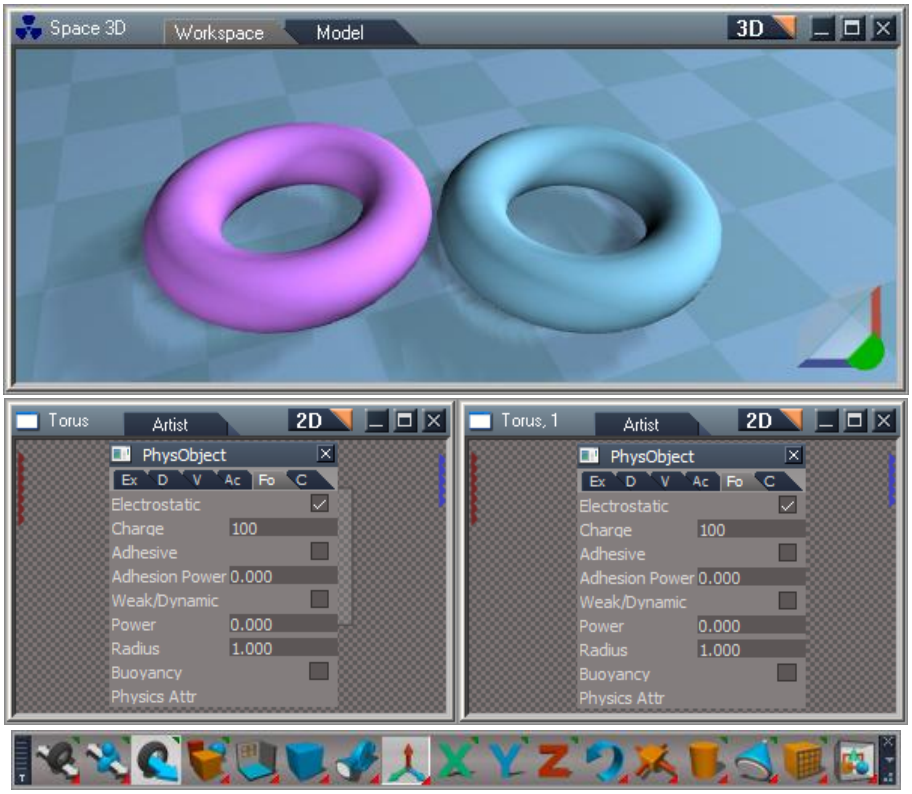
- **Adhesion Power:** set this particular value to increase or decrease the power this object's adhesion will have.
- **Weak/Dynamic:** sometimes referred to as string effect. When checked, you enable this force for the object. Weak Dynamic only works between objects with this setting enabled.
- **Power:** how powerful the Weak/Dynamic force will be for the object.
- **Radius:** distance the Weak/Dynamic force will be effective over. Distance is calculated from object's center.
- **Buoyancy:** objects with mass in a low value (.2 or so), will or are considered lighter than air so they will rise or move opposite to gravity's pull/forces on them. If you wish to have the object become buoyant, select the Buoyancy checkbox and adjust the object's Mass on default aspect in order to facilitate buoyancy.



tsPhysics1f: explore PhysObject Forces aspect: Electrostatic and Charge values

- **video link**

The Forces aspect contains values that will provoke attraction or repulsion between objects. There are three areas of the Forces aspect, covering Electrostatic charge, Adhesive power and Weak/Dynamic power and radius. In the illustration below, two torus objects have identical physical properties. When the physics simulation is started, the Electrostatic charges on each torus will serve to repel the two apart. You can enter negative numbers into the Charge value, which in this case, if only one charge value is changed, would lead to an attraction between the two objects.



Two torus objects, each with positive Charge values, will repel each other when simulation is started.

Once you begin to experiment with the values on the Forces aspect, you will begin to see the power of trueSpace at work. The values can be combined for an even larger group of possible scenarios that can be created just on this aspect of the PhysObject alone!

tsPhysics1g: explore Forces aspect: Adhesive and Adhesion Power

- **video link**

tsPhysics1h: explore Forces aspect: Weak/Dynamic, Power and Radius

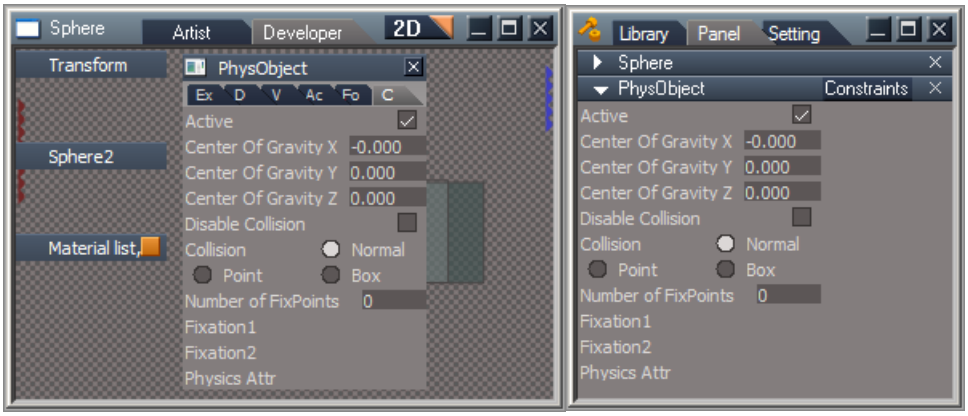
- **video link**

- Constraints

Center of gravity X, Center of gravity Y and *Center of gravity Z* allow you to modify or set a center of gravity for the object. By default the center of gravity is set to the location of the object's axis.

Disable Collision and **Collision** can be used to set special flags for the collision detection algorithm for the object. If Disable Collision is checked, collision between this object and other objects in the scene is fully disabled and only a collision with ground is detected. The object will simply pass through and ignore other objects in the scene.

The Collision attribute sets a precision of collision checks between a physical object and other objects. The default value of this attribute is Normal, meaning that during a simulation collision points are computed from all of the object's mesh intersections. This is the slowest setting, considering the number of calculations required. You can speed up the physics by setting the Collision value to either Point or Box, depending on what the situation warrants. With Point as the choice, collision between intersections of the vertices is computed, with an average point being used as a basis for the calculation. This method, although faster than the Normal setting, loses precision, so it ends up being a trade off between speed and precision. If you set the value of Collision to Box, only collisions between oriented bounding boxes of colliding objects are considered, and mesh intersections are not evaluated. Once again, this is faster than the Point setting; however precision is at its lowest when Box is chosen.



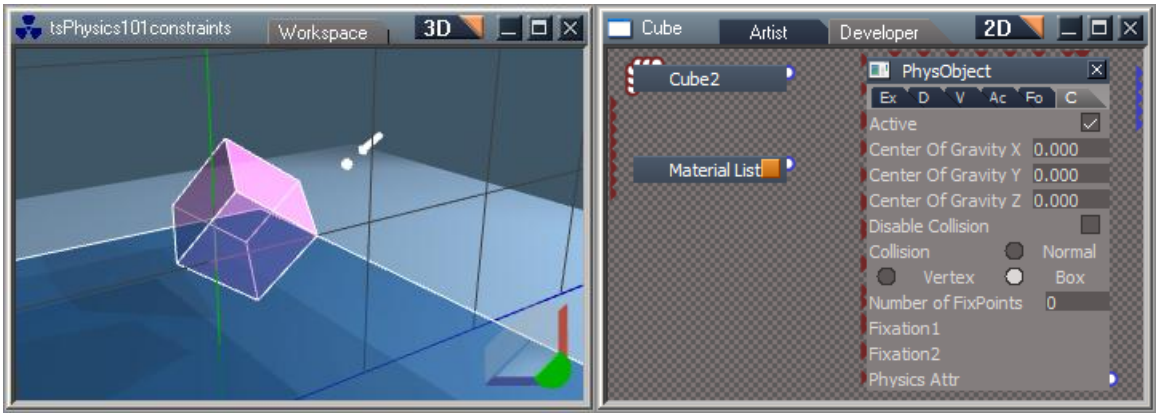
PhysObject Constraints aspect in the LE and the Stack

- **Active:** determines if the Constraints panel will have affect on the object or not.
- **Center Of Gravity X:** just as with real-world physics, each object has a center of gravity. In trueSpace, you define center of gravity by locating it using xyz coordinates. X sets location for x-axis.
- **Center Of Gravity Y:** set location of COG for y-axis.
- **Center Of Gravity Z:** set location for COG for z-axis.
- **Disable Collision:** determines if collisions are calculated for this object.
- **Collision:**
  - **Normal:** collisions between vertices, edges and faces are computed. Slowest setting because of heavy computations required for all collisions.
  - **Point:** collisions for vertices are only calculated for this object. A vertex may collide with another object's vertices, edges and faces, however calculations for this object's edges and faces are not calculated.
  - **Box:** fastest calculations as the bounding box for this object is used for collision detection. The bounding box is a cube that encompasses all the geometry of the object. Given that its structure is fairly simple, collision calculations are rapid but less accurate.

- **Number of FixPoints:** as you add fixation points to this object, the number (0, 1 and 2) are displayed here but used by trueSpace physics engine. Let trueSpace populate this value for you. No need to set manually.
- **Fixation1:** when you use the FixPoint1 tool  a new object is created inside the object. The object is named PhysConstraint. The PhysConstraint has output connector called Vector, which is connected to this Fixation1 input connector on the PhysObject.
- **Fixation2:** just as with Fixation1, using the FixPoint2 tool  a second PhysConstraint object

is created inside the object and connected to the Fixation2 connector on the PhysObject..

- **Physics Attr:** this attribute is common across all the aspects. It is an output parameter that gathers up all the information and values from the 5 aspects and packages them all up for the physics engine to utilize.



tsPhysics1i: explore Constraints

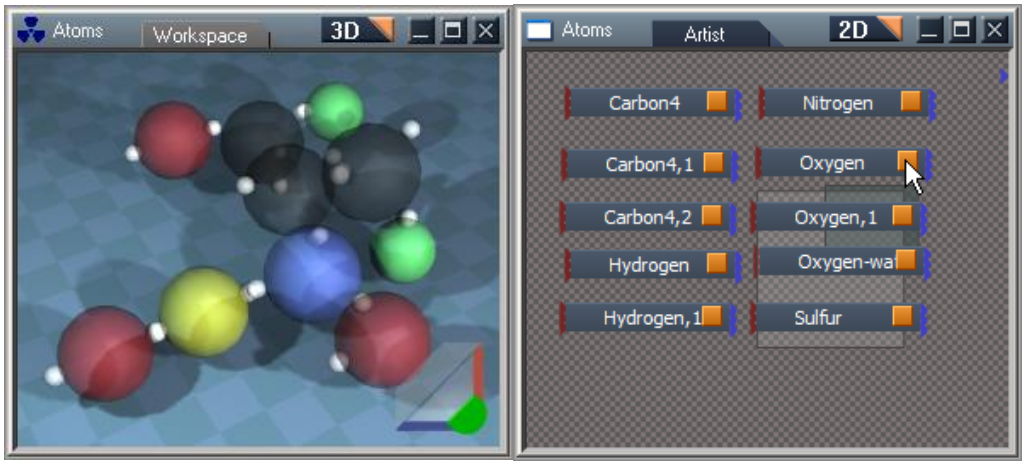
- video link

10.2.2 Sub-objects with physical attributes

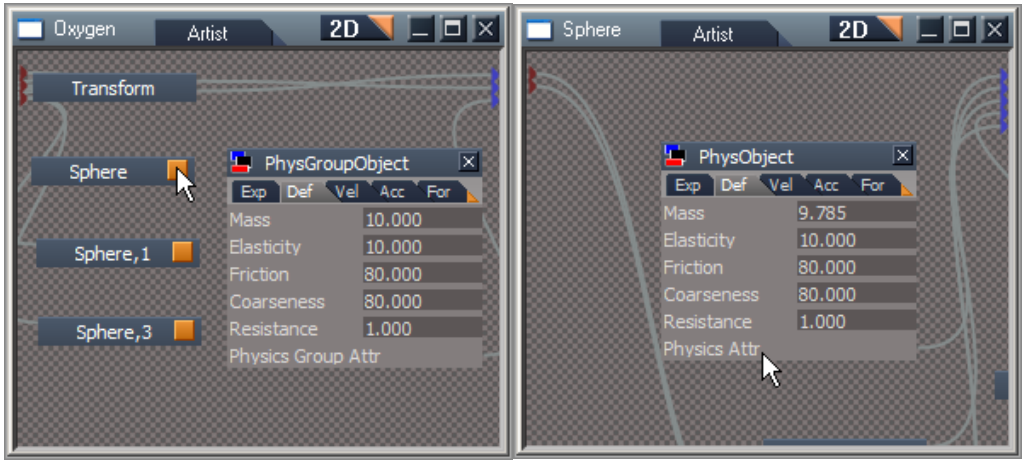
If you assign physical attributes to an object with several parts, each sub-object (that has a Mesh and Matrix) of this object has its own set of physical attributes represented by the *PhysObject* and the output *Physics Attr* (which belongs to the sub-object).

The overall attributes for the whole object are still represented by the *PhysObject* , (in the Atoms scene this is named *PhysGroupObject* the name can be arbitrary for objects) but this time the output connector is named *Physics Group Attr* and the attributes are computed from the values of the *PhysObject Physics Attr* objects that belong to the individual sub-objects.

The advantage of this setup is that in a simulation you can set different attributes for each sub-object (different elasticity, friction, weight) and simulate a complex object consisting of different materials. (For example, a car can have rubber wheels with an elasticity of 90 and an iron body with an elasticity of 40.)
The Atoms scene in trueSpace takes advantage of Weak/Dynamic forces. Each of the ten objects contain additional sub-objects which each have individual physical properties set that affect the overall dynamics of the scene. You are encouraged to load the scene from the Scenes – Base library and explore how the physics works in this scenario.

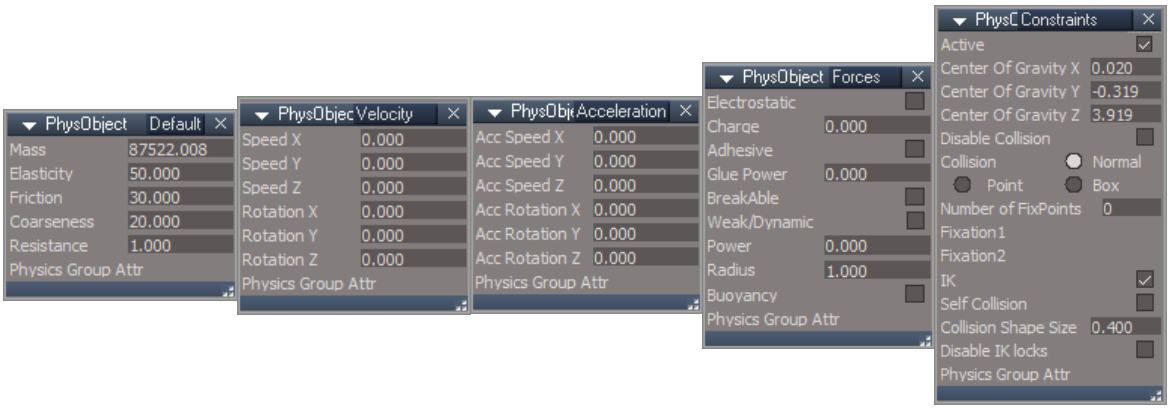


Atoms scene contains examples of group-physics at work



Inside the Oxygen atom where we see PhysGroupObject with a Physics Group Attr connector. Inside a sub-object is a PhysObject with just Physics Attr connector.

- **Physics Group Object attributes and tabs**



The properties and aspects of the Physics group object contain all the settings for the physics object , and also contain some additional physical attributes that are only relevant to a grouped object , these settings are displayed in the Forces and Constraints aspect for PhysObject.

- **Forces Aspect Breakable Objects**

The breakable option can be set only for objects that consists of more than two parts.
The additional Forces attributes for the Physics group object are Glue Power and Breakable.

- Forces:

Forces aspect in the PhysObject for an object with Physics Group Attr
Left Physics Group object has some additional attributes, Right single object Physics Attributes

If the Breakable checkbox in the PhysObject panel is checked, an object can break into smaller parts during the simulation when colliding with other objects or the ground.
Breakable is used in conjunction with the Glue Power attribute which modulates the intensity of a required collision impulse to break some parts away from the object, or more exactly it gives the magnitude of force required to break

the object apart. The higher the Glue Power value, the more intensive object-object or object-ground collision impulse required to break the object. If Glue Power is zero, an object will break at any collision events, eg: just by running the simulation will allow the object to break apart .

- Glue Power.
  - The amount of force needed to break apart the grouped object . Low values mean the object breaks easily on collisions , high values means it takes more force from a collision to break the object apart.
    Valid Values range from 0 to 100.

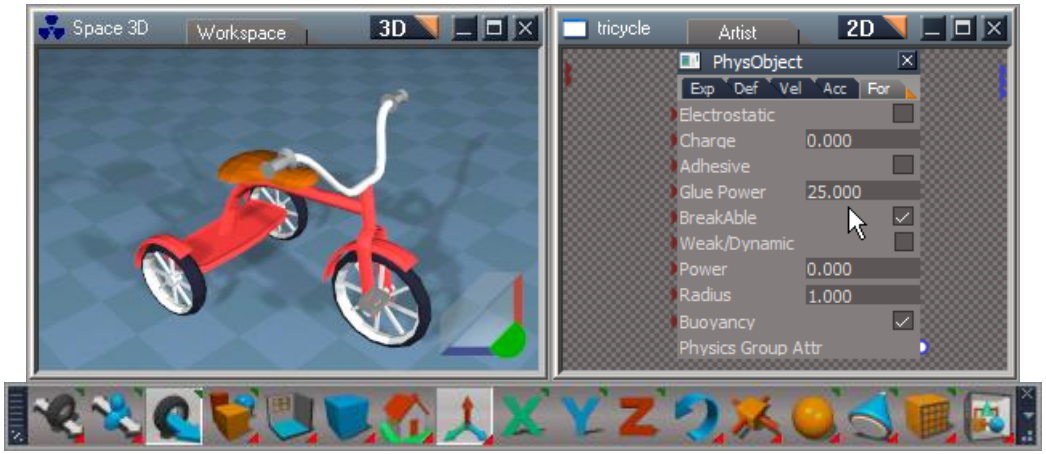
- Breakable .
  - If checked the grouped object can break apart. Each sub-object then is able to participate and react as an individual during the simulation and the surfaces of the sub-objects will participate in collision events.
  - If unchecked the grouped object behaves as if it was just a single object and doesn't break apart with collision events during the simulation.

➢ Example: The tricycle from the Objects Base Library is a good object to use to experiment with the Breakable and Glue Power values.

Load the tricycle, add physics attributes to it  enter the tricycle, change the PhysObject for the tricycle to the Forces aspect, and place a check in the BreakAble checkbox.
Run the simulation and observe the results. Since Glue Power is 0 the tricycle breaks apart.

Tricycle breaks apart during simulation with Glue Power set to 0

Change the glue power to 25 and run the simulation again. The tricycle appears to behave as if it was made from one solid object. This is because no impacts or collisions are occurring and the glue power is too great to allow the object to break apart on its own forces.

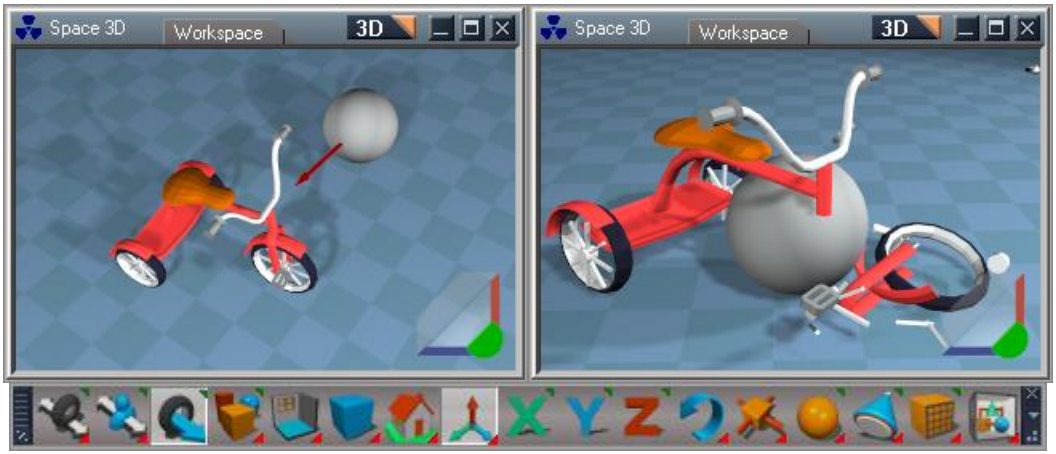


Tricycle appears to be stuck together during simulation with Glue Power set to 25

Run the simulation again and this time Select the Phys Move tool ,raise the tricycle a little off the ground and release it. The tricycle will begin to break apart during an impact or collision with another object or the ground.

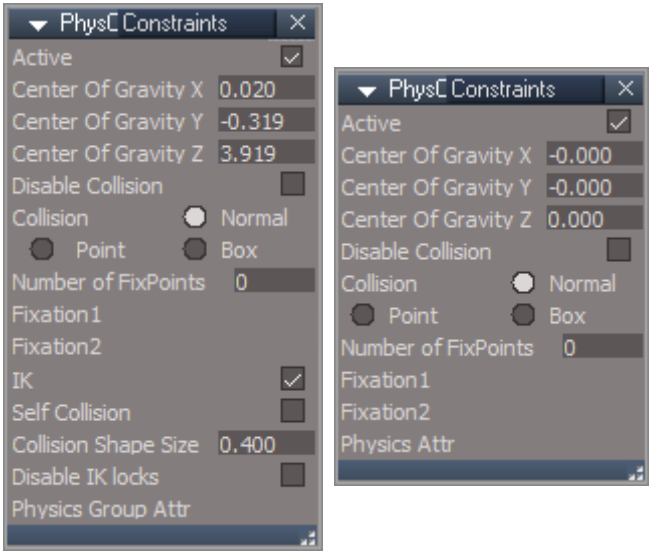
tricycle Breaks Apart on collisions with the ground during simulation with Glue Power set to 25

When another object collides with a physical object with Breakable checked the collision affects the grouped object and breaks it apart on impact during the simulation.



tricycle Breaks Apart on collisions on objects impact during simulation with Glue Power set to 25

- **Constraints Aspect: Characters, IK, Self Collision, Collision Size, Locks.**



Left Physics Group object has some additional attributes, Right single object Physics Attributes

The additional attributes in the Constraints aspect for the Physics group object are IK, Self Collision, Collision Shape Size, Disable IK locks, these settings are mostly applicable to characters and their reactions during a simulation. By default, characters collide with all objects in the scene and collision between a character's parts are ignored (bones could go through each other).

- Reference Characters and Physics

- IK
  - Checked : IK for the character joints and bones is considered during the physics simulation.
  - Unchecked : the character will behave like any ordinary solid group object without considering its joints and bones during the physics simulation.
  - Note: This value cannot be changed during simulation and needs to be set before running a simulation: an object can either have character-like or solid object-like behavior during a simulation.

- Self Collision
  - When Checked will enable collision between parts of a grouped object or if a character object the bones collision shapes. Self-collisions can be turned on or off during the simulation.

- Collision Shape Size
  - set or modify the thickness of Collision shape size of bones or objects. A Default value of a shape size is 0.4. For example with characters if you increase the thickness of bone's shape, the collision will be detected further away from the bone or object. You can interactively change this attribute during simulation.

- Disable IK locks
  - When you edit a pose of a character for example with Dynamic Pose tool, you can use many IK locks (rotation, full, position). To enable these locks in the simulation, check the Disable IK locks attribute. By default, all locks are disabled. Enable or disable of locks can be done interactively during the simulation (toggle on/off).

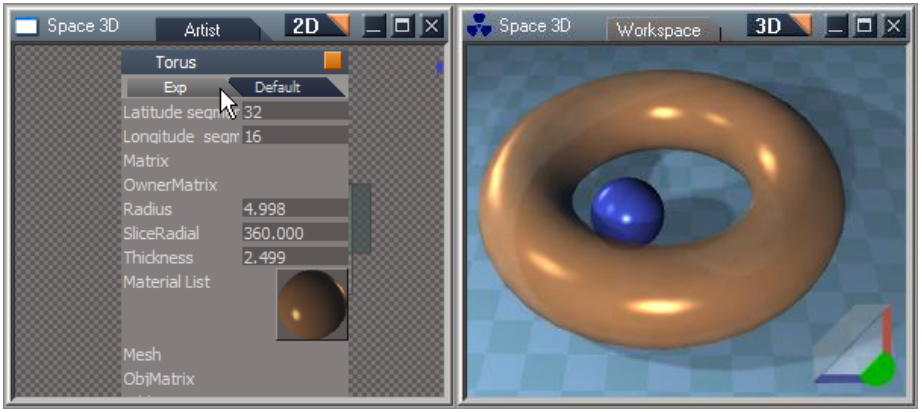
➢ Reference Characters and Physics

Command - Disabling of collisions for static objects

Collision detection can be disabled for static objects in the scene. This allows having static objects in the scene (simulation) that do not interact with objects with physical attributes.
You can utilize this feature also for FPN collision, when you wish to turn off collision for certain objects. To change behavior of object in collision detection use a command with following syntax:
Physics.Collision*(bEnable, objname)*
where ***objname*** is full name of object for which you wish to change collision detection (for example /Project/Space 3D/Sphere) and parameter ***bEnable*** can be false if you wish to disable collision and true if you wish allow collision.

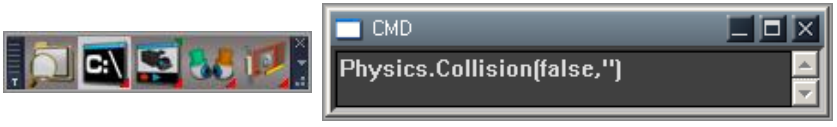
The command can also be applied for the currently selected object in the scene using the following syntax: ***Physics.Collision****(bEnable,'')* in this case, the collision will be enabled (***bEnable*** = true) or disabled (***bEnable*** = false).

Example: The Sphere has physics properties, Torus does not have any but when the simulation runs the Sphere will collide with the Torus and cannot pass through its geometry boundary.

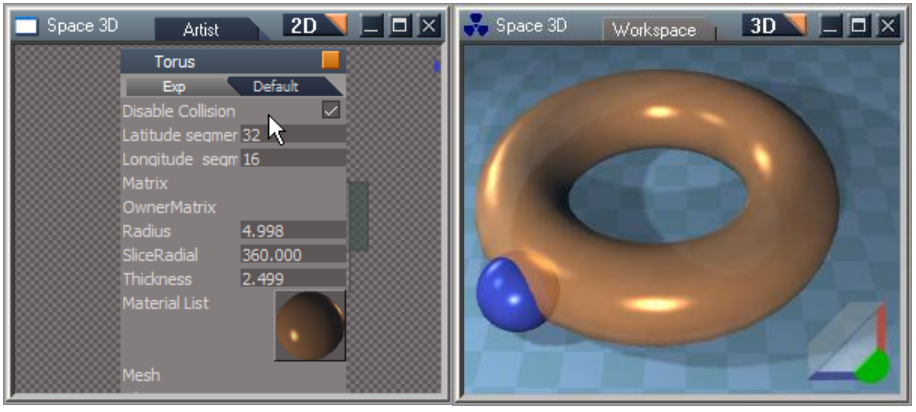


Sphere cannot pass through Torus when simulation is run

By opening a command window and typing: Physics.Collision(false,") and pressing enter an extra attribute called Disable Collision is created on the expanded aspect for the selected object which can be toggled on and off if desired later to disable or enable collisions with the object . Now when physics is run the Sphere no longer collides with the Torus and passes through it.



Command window

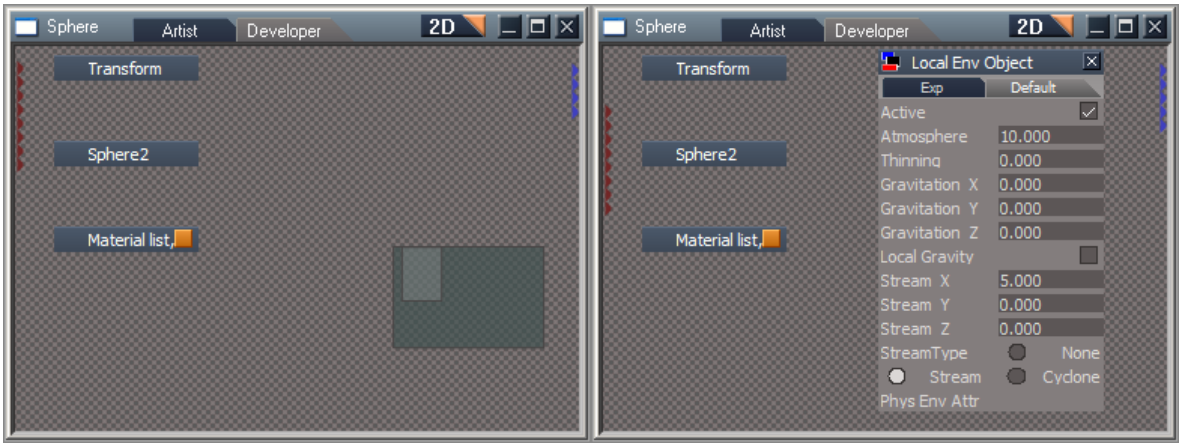


Attribute created by the command to disable collision, sphere passes through torus

10.2.3 Local Environments - Liquids and Gasses

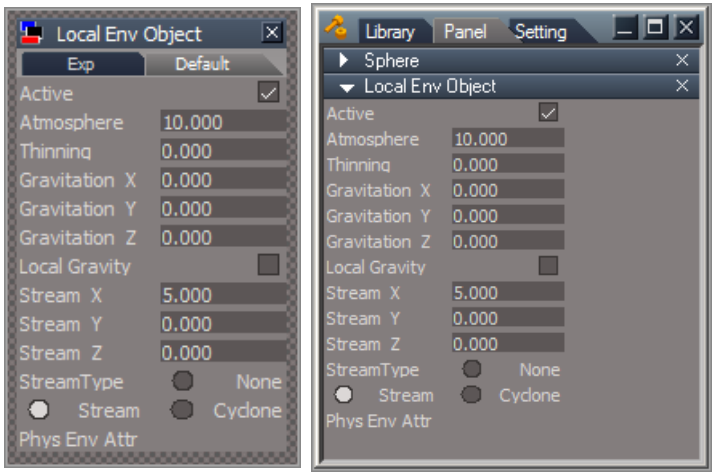
Add Local Env:

L-Click on the tool creates a local environment within the confines of the object's geometry. When you add a Local Environment to an object, LocalEnvObject is created inside the object. Default values are used for the attributes and can be edited either in the panel in the stack or by using the Link Editor to navigate inside the object to access the LocalEnvObject object and parameters it houses.
R-click on the tool will show the selected object's Local Environment settings in the stack if the object has this property set.



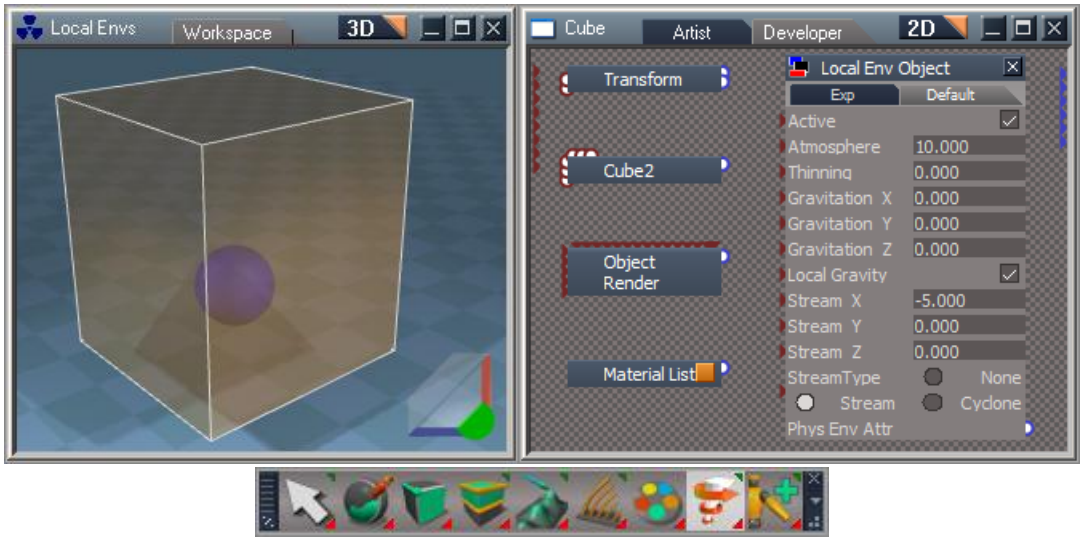
Add Local Env Object

- LocalEnvObject attributes



Default Local Env Object in LE and Stack

- **Active:** Determines if the Local Env Object will be active in the simulation or not.
- **Atmosphere:** Set the atmosphere density with this value.
- **Thinning:** height within the local environment where the atmosphere thins; has lesser effect on the objects inside. Most powerful area is at base of object and as other objects rise/move away from the base, the atmosphere has less effect. Just as earth has an atmosphere that thins as you move away from the earth itself.
- **Gravitation X:** Set gravitation force for the X axis.
- **Gravitation Y:** Set gravitation force for the Y axis.
- **Gravitation Z:** Set gravitation force for the Z axis.
- **Local Gravity:** Set whether the Local Environment will have gravity during the simulation, or if left unchecked, the scene gravity will prevail.
- **Stream X:** Set power for stream force in X direction.
- **Stream Y:** Set power for stream force in Y direction.
- **Stream Z:** Set power for stream force in Z direction.
- **StreamType:**
    - **None:** stream is turned off.
    - **Stream:** stream force has linear direction/force.
    - **Cyclone:** stream force has a circular force/direction.
- **Phys Env Attr:** all the values for the settings on the Local Env Object are bundled together and exported to the physics engine for computation in simulation.

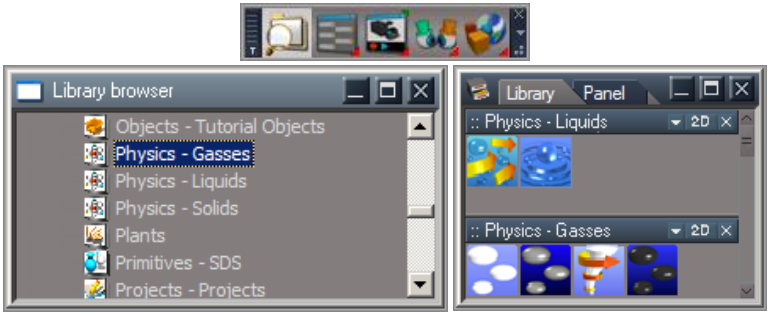


tsPhysics1j: explore Local Env Object

- **video link**

Objects in the Liquids and Gasses Libraries represent Local environments. The pre-sets are; Air, Space Vacuum, Tornado, Vacuum, Water Stream and Water. Each of these pre-set local environments are based on the settings found on the Local Env Object.

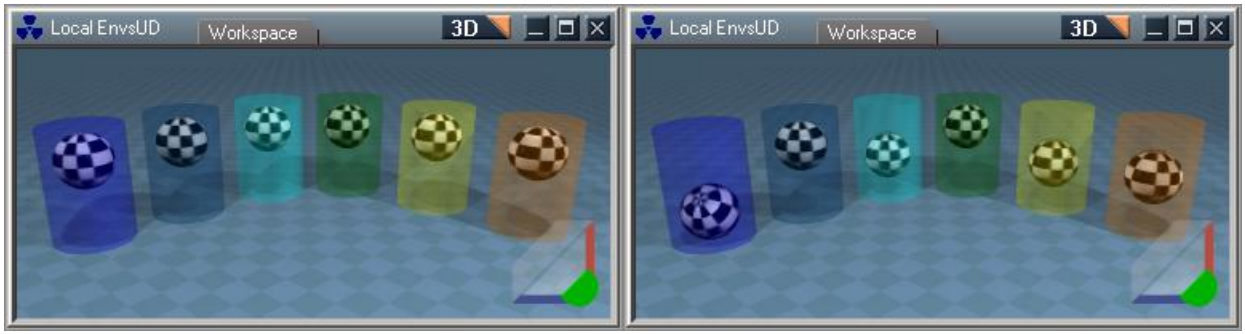
These items are used for the simulation of various physical environments that can be present in a scene/scenario. Local Environments are represented by the physics environment object. The placement of the environment object within some object (eg:cylinder) assigns local environment attributes to this object, and the properties of the local space will be valid in the volume/confines of the object.



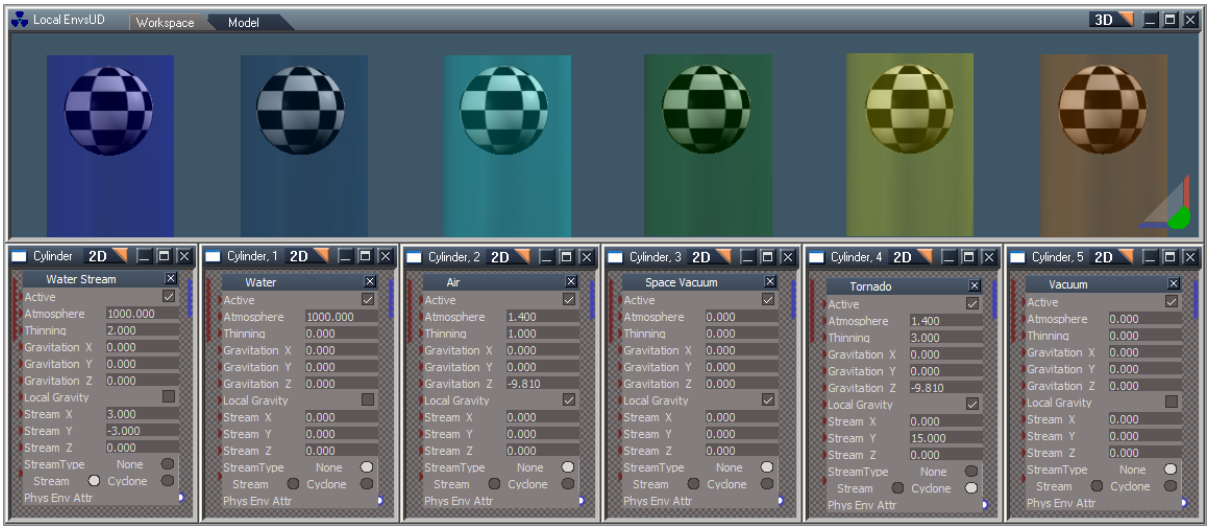
Physics- Liquids and Physics – Gasses Libraries

Example scene : 6 cylinders, each with a different environment, are placed in the scene. Then a material is applied from the Physics- Liquids and Physics – Gasses Libraries, onto each of the individual cylinders .

The checkered ball is assigned physical properties, then copies made and placed in each of the cylinders.



Time Lapse example of the six different environments



Setup an experiment to review local environments

When the simulation is run, you can watch the reaction of each sphere/ball inside each cylinder. The Water Stream is interesting in this scenario, where the force of gravity on the checkered-ball is moving against the water stream's direction. As the forces reach a point of equilibrium, the ball begins to move upwards (downstream) and float on surface.

With the help of the local environment object you can define the local atmosphere density, local atmosphere thinning, local gravitational forces, and two kinds of moving environments: **stream** (movement in one direction), or **cyclone** (movement is circular). If the StreamType is set to "None", streaming is deactivated. With streaming enabled, any objects within or entering that local environment will be affected by the forces within that environment, as if they were being carried along in a stream or caught up in a tornado.

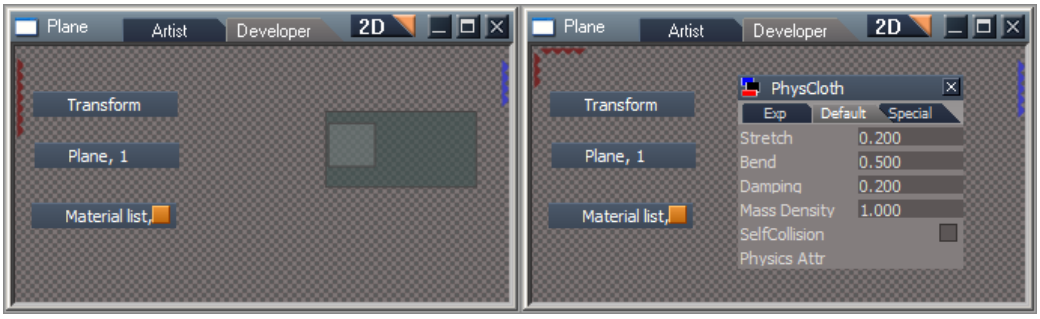
Local Environments only affect objects with Physical Attributes applied; that is, they only affect "active" objects.

10.2.4 Physics Cloth

Add Cloth:

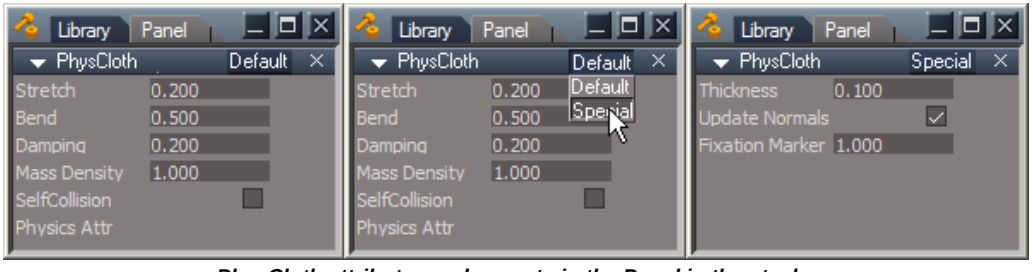
Adds the cloth attributes to an object. First select the object and then L-click on the Cloth icon:

Physical cloth attributes can be assigned to any object that has a Mesh and Matrix. The current implementation doesn't support hierarchical objects (with more sub-objects); only simple objects can have cloth attributes assigned. The cloth attributes are represented by the *PhysCloth* object, which is created inside the object to which it was applied .
When you use this tool to add physics Cloth properties to a mesh it will create a PhysCloth object.

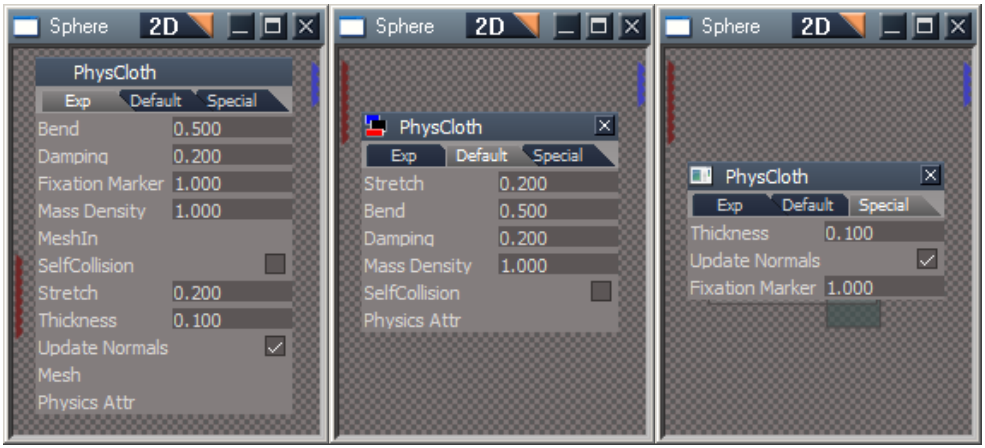


the PhysCloth object

The properties for the cloth can be set or adjusted by using the panel in the stack which is visible when the cloth tool is used or from the LE inside a mesh that has had cloth applied.
The stack will only show the aspects for the Default and the Special whereas in the Link editor you can see the expanded aspect which shows all the available attributes on one aspect.
Tip:- If you wanted to see the Exp aspect in the stack you can R-click in the title bar in the LE and choose Detach from the menu.



PhysCloth attributes and aspects in the Panel in the stack



PhysCloth attributes and aspects in the Link Editor

The available attributes for PhysCloth which can be adjusted are described below:

- PhysCloth object attributes

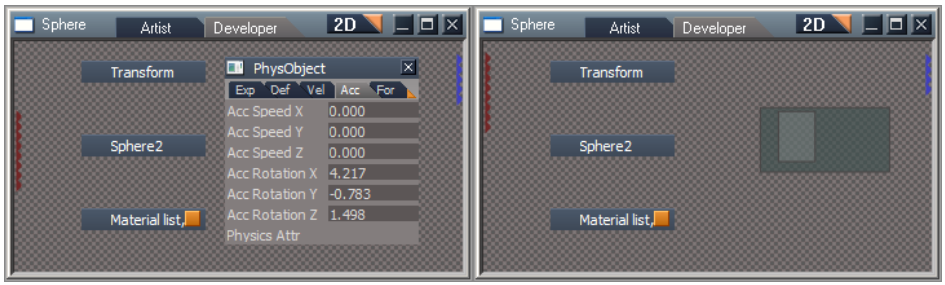
- **Bend:** specifies the rate of allowed bending between neighboring triangles of cloth mesh. The values can be in the range from 0 to 1 – the value of 0 means no bending allowed, whereas value 1 means maximum bending allowed.
- **Damping:** specifies the rate of damping of vertex movements in the cloth mesh. The values can be in the range from 0 to 1 – a value of 0 means no damping, a value of 1 means maximum damping (in this case cloth should behave more like a solid object for example).
  Fixation Marker: specifies the size of fixation points if cloth fixation tool is used with this object (widget tool reads this value and sets the size of fixation markers accordingly).  Cloth fixation tool
- **Mass Density:** specifies density of cloth – the total mass of cloth object is product of cloth's surface and mass density.
- **Self Collision:** If Selfcollision attribute is checked, self-collisions between cloth mesh triangles are considered during simulation.
- **Stretch:** specifies the rate of allowed stretching between neighboring vertices of cloth mesh. The values can be in the range from 0 to 1 – a value of 0 means no stretching allowed, a value of 1 means maximum stretching allowed.
- **Thickness:** define the thickness of cloth considered in collisions evaluation.
- **Update Normals:** when checked updates the mesh's normals if they are present.
- **Physics Attr:** all the values for the settings on the PhysCloth Object are bundled together and exported to the physics engine for computation in simulation.

Limitations- Cloth simulation cannot be captured to keyframes for rendering of animations.

- Example Scene : 3 Flags : provided by Stephen Beauchamp.
- Instructions : Load the scene , Run Physics. Flags Flap in the wind.
- Resources contains some additional flags to change flag textures.

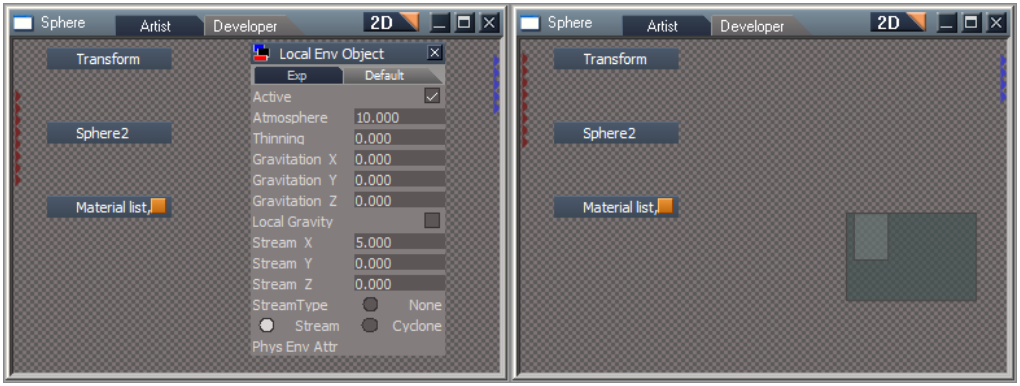
10.2.5 Remove Physics properties

Remove Phys Attrib: Left-Click will remove/delete the PhysObject object from within the selected object.



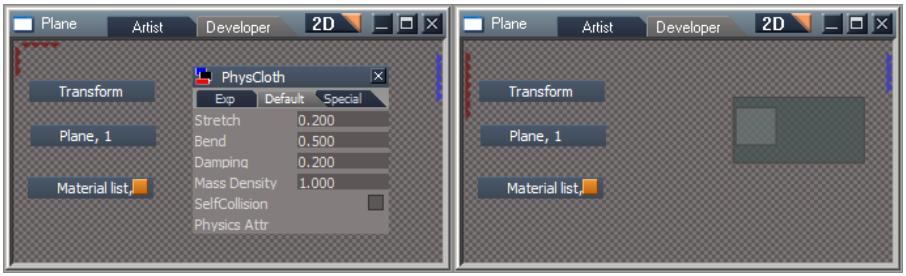
Remove PhysicsObj

Remove Local Env: Left-Click will remove/delete the LocalEnvObject object from within the object.



Remove Local Env

Remove Cloth: Left-Click will remove /delete the cloth attributes from an object.



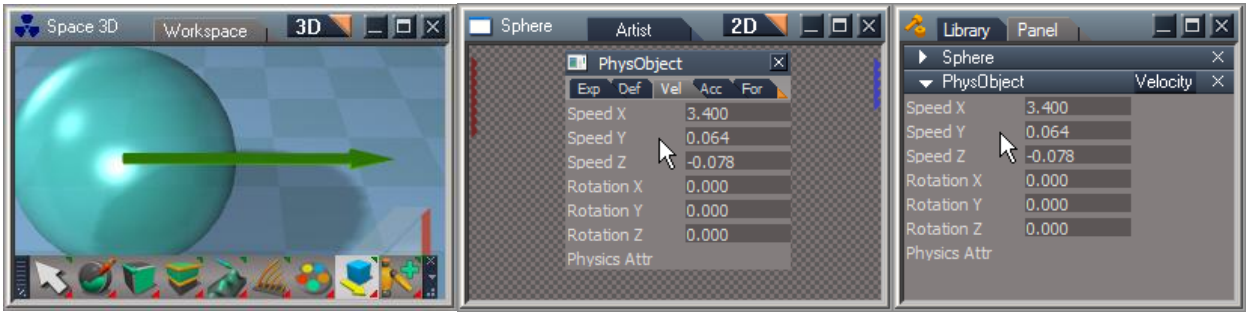
Remove Cloth

10.2.6 Speed and Rotation

Speed and Rotation tools :

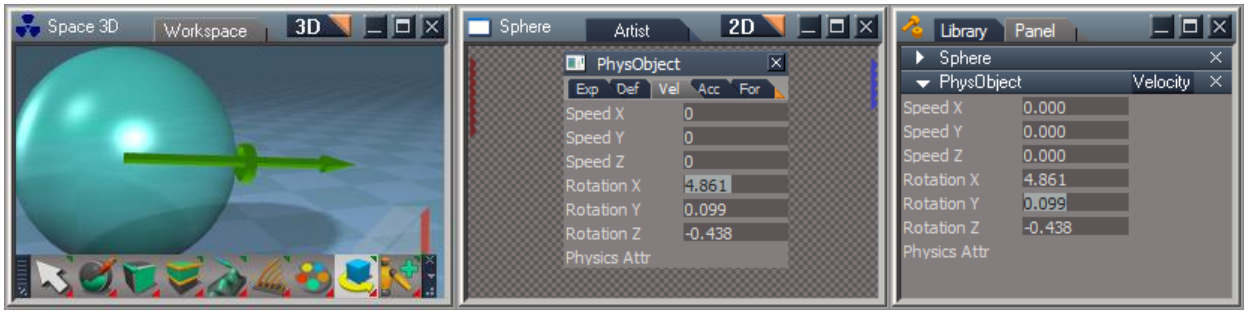
There are four tools for modifying the initial kinematics attributes with physics: L-click on the tool shows a widget in the workspace which can be manipulated by dragging on its surface for direct manipulation and the corresponding values will be applied to the PhysObject properties for the selected object. The values the physics widgets relate to can also be manually input into the corresponding attributes for the selected object in the Panel in the Stack or can be located and input directly in the Link editor for the object.

Speed : Adjusts the values for Speed X, Y, Z in the Velocity Aspect.



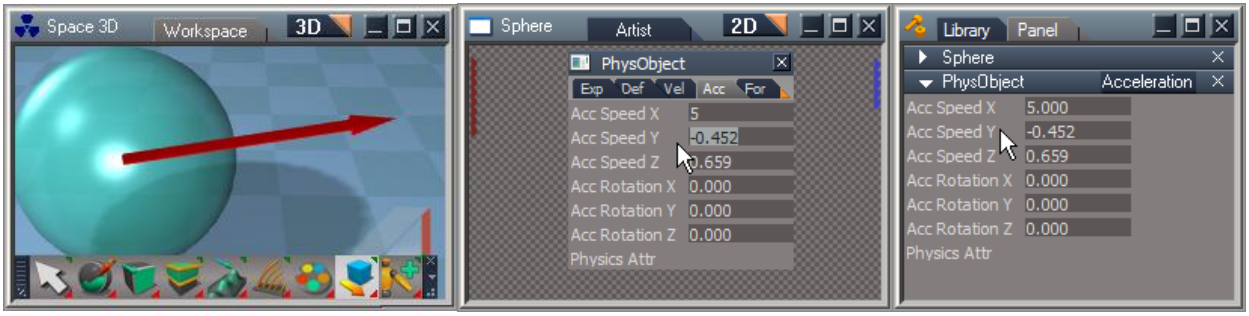
Widget for Speed X, Y, Z in the Velocity Aspect.

Rotation: Adjusts the values for Rotation X, Y, Z in the Velocity Aspect.



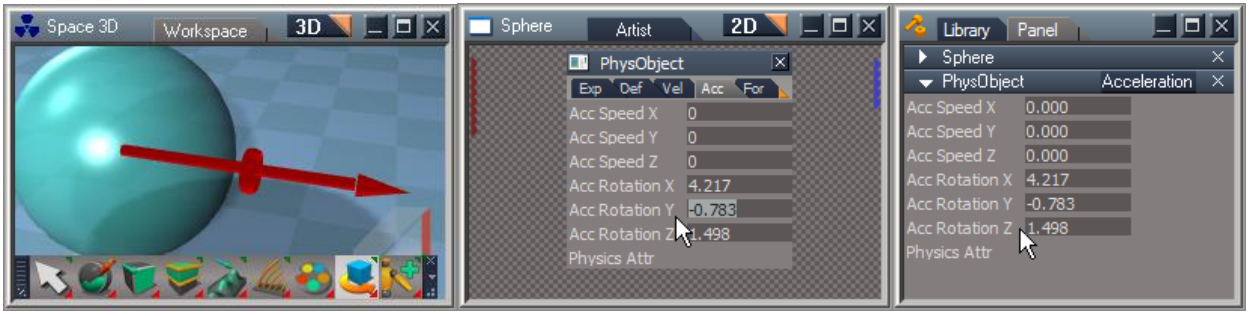
Widget for Rotation X, Y, Z in the Velocity Aspect.

Speed Acceleration: Adjusts the values for Acc Speed X, Y, Z in the Acceleration Aspect.



Widget for Acc Speed X, Y, Z in the Acceleration Aspect.

Rotation Acceleration: Adjusts the values for Acc Rotation X, Y, Z in the Acceleration Aspect.



Widget for Acc Rotation X, Y, Z in the Acceleration Aspect.

Tip:  Transparent Modes.

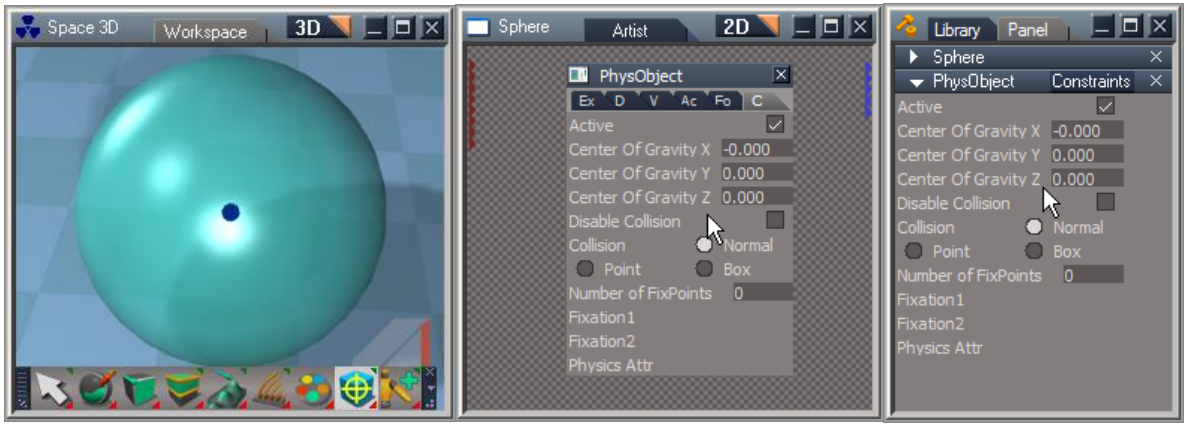
When working with and setting up physics objects a good setting to use is one of the transparent view modes for the

Workspace window as it can help to see and manipulate the physics widgets more easily.

10.2.7 Centre of Gravity and Fixation

COG and Fixation tools

Center Of Gravity: L-Click on the tool shows center of gravity for the object and is adjusted by moving the widget's "blue dot" in the Workspace or by input in the Constraints aspect in the LE, or the panel in the stack.

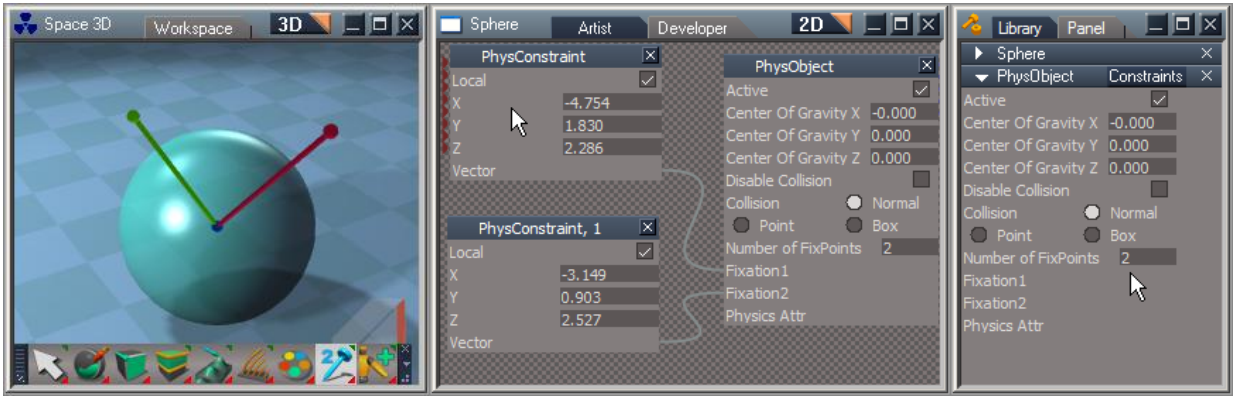


Widget for Center Of Gravity in the Constraints Aspect

Fixation point 1: adds fixation point - Green widget can be used to adjust values or values can be manually set in the LE for the PhysConstraint object.

Fixation point 2: adds fixation point - Similar to fixation 1 both fixation points can be adjusted by moving the Red widget in the Workspace or by numeric input in the LE for the X,Y,Z attributes of the PhysConstraints objects which are created when the tool is used on an object which has physical properties already assigned to it. The stack does not have access to these objects and only shows the *Number of FixPoints* used.

Every active physical object can be fixed by one or two fixation points called Nails. The position of nail points is handled by the attributes *Fixation1* and *Fixation2* in the PhysObject .



Green fixation point 1 Red fixation point 2. PhysConstraints in the LE, Panel shows fixation points used only

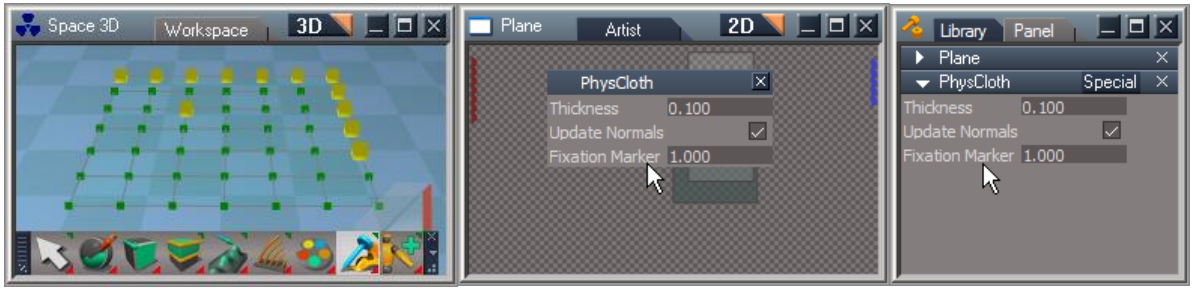
The image below, shows the result of selecting the "Fixation1" tool. This creates the PhysConstraint object in the LE and displays the "Nail" controller in Workspace View. Simply place a Fixation/Nail on one of the torus objects and in Workspace, move the nail a little. Run simulation and use the PhysMove tool  , to move the torus ever so slightly. You will see that the torus reacts as if tethered/connected to a rope. It can move around in a circular fashion within the radius of the nail's length.

A close-up when adding fixation points to an object

In the scenario illustrated below, second fixation point/nail is added to the object. This creates a second PhysConstraint object in the Link Editor. The scenario will work with our PhysEngine setting for xy plane physics, however if you set the PhysEngine to use "None" for plane physics, we can see clearly how the second fixation point is restricting the torus' movement in Workspace. Experiment with moving the controllers and re-start the simulation to review the changes in how the simulation plays/runs.

Using both Fix1 and Fix2 will result in a different reaction to your Physics Scenario

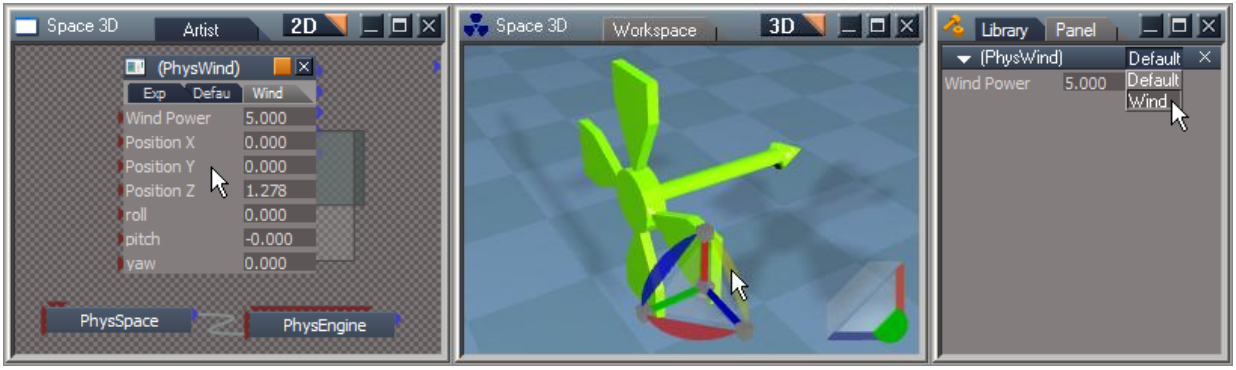
 **Cloth Fixation:** adds fixation points to physics cloth objects. L-Click the tool then L-mouse press and drag over the points to be fixed. L-Click and drag on existing fixation points will remove them. R-Clicking over the tool while it is active will remove all fixation points on the currently selected object. Adjust the size of the fixation points in the Fixation Marker in the Panel.



10.2.8 Wind and Environment

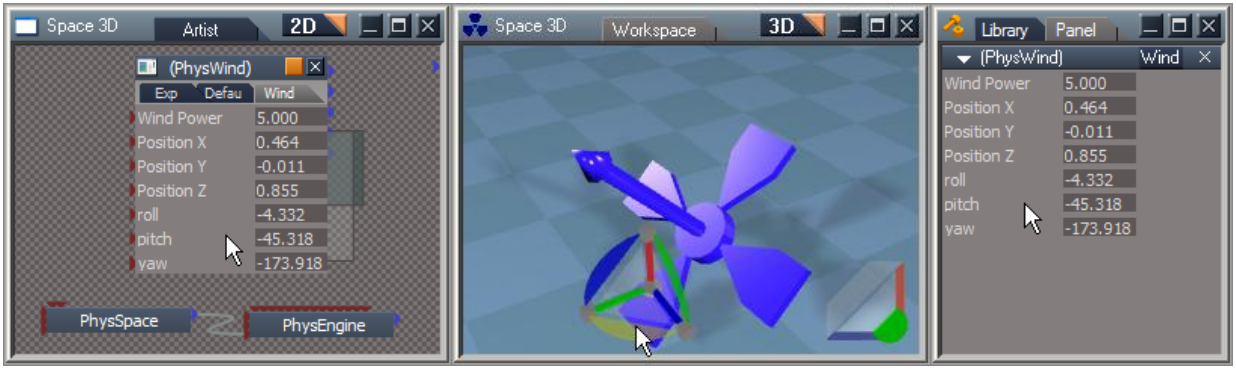
Wind and Environment L-Click on either of the Wind tools creates an object to represent the wind in the Workspace view. R-Click will show the panel in the stack for the wind object. The wind object position and direction are set by using the object transform widget: as you move the widget surfaces the panel's values update. The strength of the Wind is set in the Wind Power in the Panel or in the Link Editor.

Edit Local Wind:



Wind tab of Local Wind in Link Editor at left, while Local Wind controller is showing at right

Edit Global Wind:

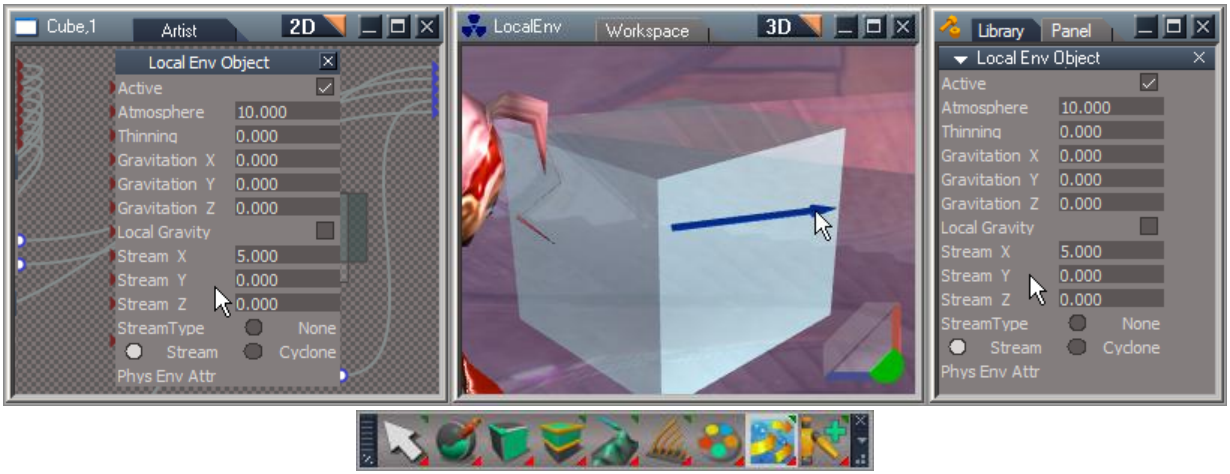


Link Editor view of Global Wind on left, with Global Wind controller on right

Edit Local Environment: L-Click on the Edit Local Environment tool shows a blue arrow controller in the Workspace view. R-Click will show the panel in the stack for the selected object if it has Local env attributes set. Dragging the widget with the mouse changes the values for Stream X, Y, Z on the Local Env Object. It offers an alternative method to directly inputting the values in the Panel or in Link Editor. To exit the tool use a R-click in the Workspace or choose another tool.

Below shows a cube with Local Environment added, and the cube is entered in the LE to show the LocalEnv object panel. When you use the Edit Local Env icon a widget shows; the blue arrow can be dragged with the mouse. As you move the blue arrow in the Workspace the values for Stream X, Y, Z will update in the Local Env Object.

Note - Objects must have Local Env Attributes  added to them first for the widget to show.



Default tab of Local Env Object at left, while Edit Local Environment controller within a cube on right

Edit Gravity: L-Click on the Edit Gravity tool shows a red arrow controller in the Workspace view.

Dragging this widget with the mouse changes the values for Gravity on the PhysSpace object, it offers an alternative method to directly inputting the values in the Panel or in Link Editor.

As you move the red arrow about the values for Gravitation X, Y, Z will update in the PhysSpace object. To exit the Edit gravity use a R-click in the Workspace or choose another tool.

Note - Physics must be enabled in a scene by using the start/stop simulation for the widget to show.

Edit Gravity tool brings the controller's red Arrow into view. Attributes for Gravity exist on the PhysSpace object

---

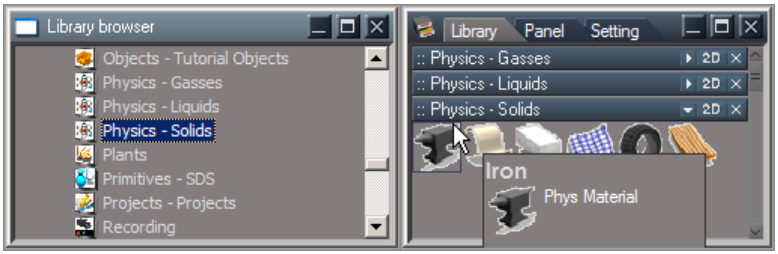
10.2.9 Libraries: Loading and Saving physics materials

The physical attributes of objects and local environment can be saved to separate files with extension:

- **.RsPhysMat** for physical attributes.
- **.RsPhysEnv** for attributes of local environments.

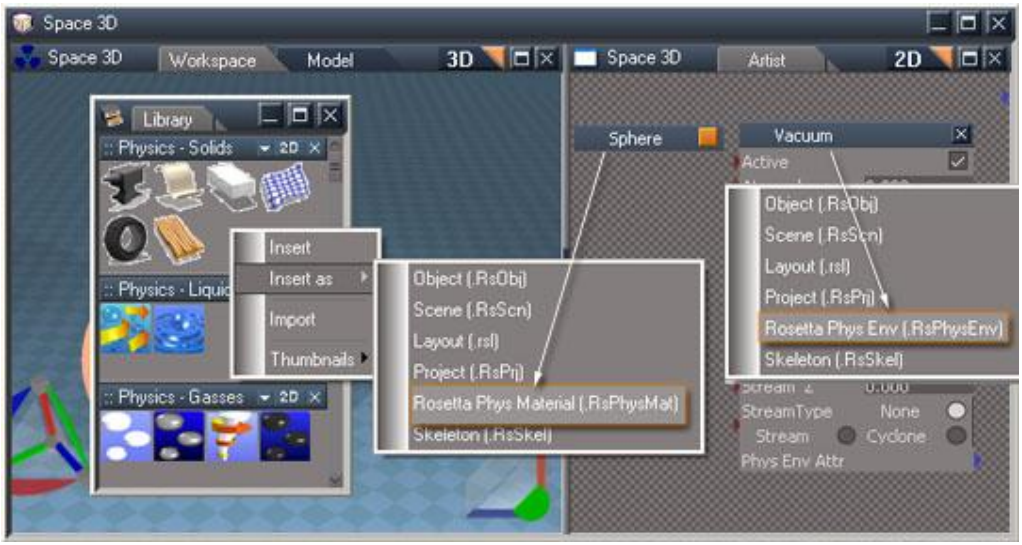
In the library system there are three libraries – Solids, Liquids and Gasses, which contain several samples of physical attributes (Solids) and local environment attributes (Liquids and Gasses). You can use these libraries to load particular attributes (item in library) to selected objects in scene. When more than one object is selected, the library item is loaded and applied to all objects in the selection. Physical attributes are not loaded for objects that can have no physical or environmental attributes (in general objects without mesh and/or matrix attributes). If objects in the scene have no physical attributes before loading (they were *static* in terms of simulation), after loading they turn into *active* objects and if simulation is running, they are immediately included in the simulation.

library browser icon



Double click the Physics Libraries, Gasses, Liquids , Solids in the loads them in the stack

To save attributes into the library, first you have to select object(s) in the scene. If selected objects have physical attributes or they are a local environment when the library insert action (right-click in library) is performed, the option for saving of the physical materials or physical environments will be listed in the menu. Use Insert as/Rosetta Phys Material from menu to save the selected object's physical attributes.



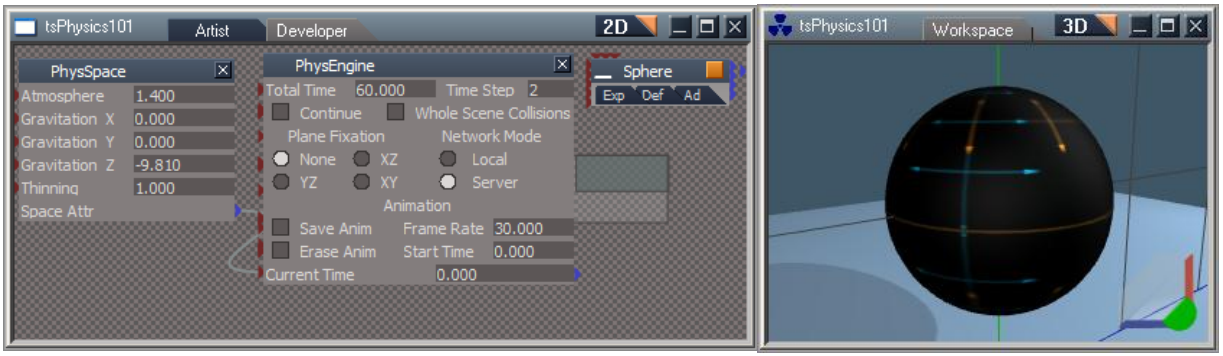
Menu offers extension for saving/inserting file, depending on what is selected

➢ Back to Section

10.3 Tutorials:

10.3.1 Tutorial: Putting it all together

The tsPhysics101 scene consists of a standard default Workspace sphere. It has a unique texture on it created to allow you to see subtle movement of the sphere while running physics simulations.



tsPhysics101 test scene

The tsPhysics101 scene uses default values for the PhysSpace and PhysEngine. The sphere itself does not have any Physical Attributes applied to it. First order of business is to select the sphere in Workspace or Link Editor and use the Add Phys Attr tool  on the sphere to create attributes for it and make it ready to use in testing this scene.

We will use default values for the sphere's newly created PhysObject object.

With the sphere now part of the physics scenario in trueSpace, activate the simulation by left-click on the Start/Stop physics tool  and as one would expect, the sphere falls towards the ground plane as it would in real-world. Left-click the Start/Stop tool again to stop the simulation and return the sphere to its original location.

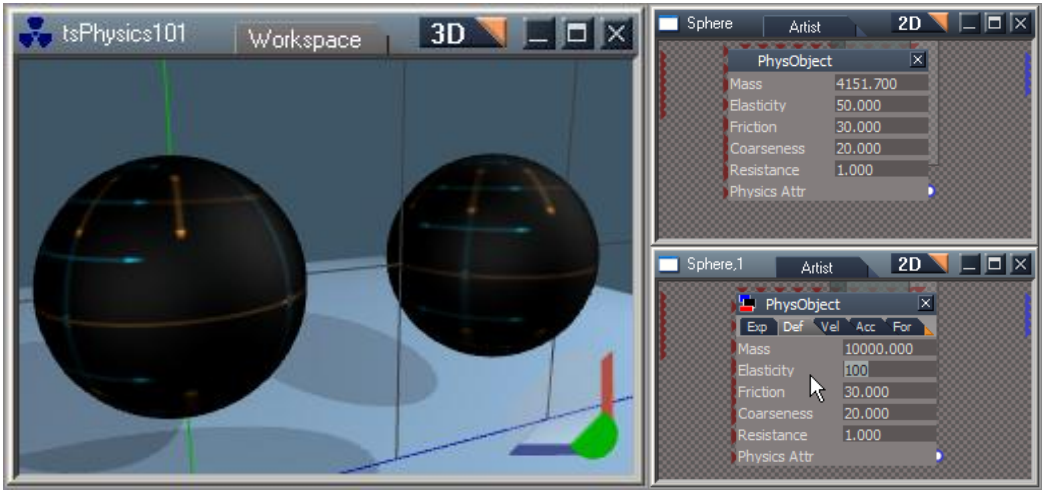
Activity 1:

- On the PhysSpace object, change the Gravitation Z value to 0.00 and also change the Atmosphere to 0.000. Once you have done this, start the simulation. You should see no action at all as the sphere sits in space. Stop the simulation.
- On the PhysSpace object, change the Gravitation X value to 10.000 and start the simulation. You will see the sphere begin to move in the positive X direction rather quickly. Stop the simulation. Repeat this again, using Gravitation Y and Gravitation Z. What happens when you use negative numbers instead of positive numbers? What happens when you use combinations of Gravitation for XYZ in both positive and negative numbers? Mix up the values and experiment with them until you have a good understanding of how to control the movement of the sphere in space.

- Lets add the PhysEngine's settings into the mix now. With your knowledge of how to control the sphere in space using Gravitation values on the PhysSpace object, begin to mix these settings with (only) the Plane Fixation setting on the PhysEngine object. If physics is restricted to only certain planes, how does this affect how the Gravitation values are affecting the sphere? You should see that restricting Plane Fixation also restricts gravitation. It is important that you understand that one setting will have an effect on other settings.
- Open up a separate AE window and examine what happens when you check Save Anim on the PhysEngine object. Start and stop the simulation to see what is being saved in the Animation Editor from a keyframe perspective. When you check the Erase Anim on the PhysEngine object, how does this affect the AE keyframes?
- Toggle the Continue checkbox on the PhysEngine object and start/stop the simulation several times. What do you see happening? Understanding how these settings work with physics will give you a deeper level of understanding as to what is happening within the physics environment in trueSpace.

Activity 2:

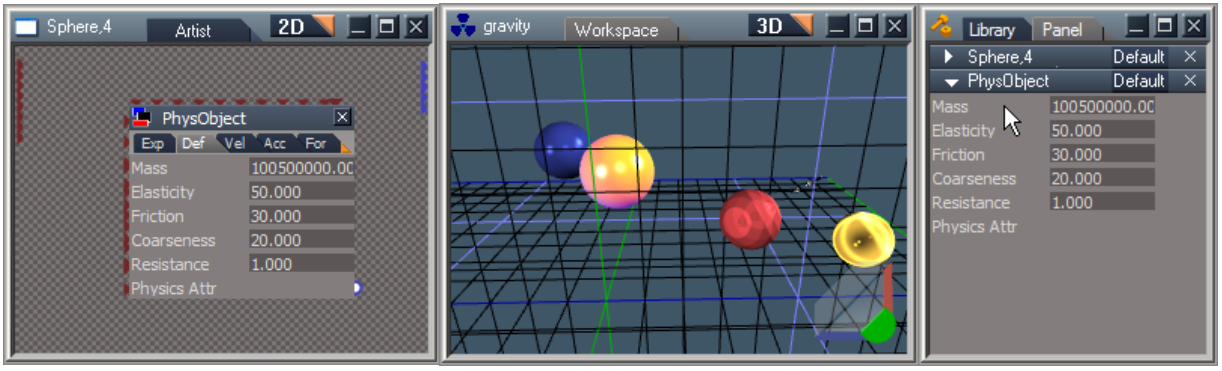
- In the Link Editor, enter the sphere object and focus in on the PhysObject inside. Up until now we have used default settings here and changed nothing. The basic intent of Activity 2 is to have you systematically go through the various aspects of the PhysObject object inside the sphere, making changes and running the simulation to see what happens. To do this effectively, you should make a copy of the sphere object and move the copy to the left of the original sphere. This copy should be done before you begin adjusting values on the various aspects of the PhysObject object.
- With the two identical spheres in Workspace, lets set the PhysSpace object back to its default values, where the Atmosphere = 1.40, Gravitation for Z = -10.80 and the rest of the values we did not really change. You could reload the scene and start from there if you wish. As long as you have two spheres in a default scenario of the tsPhysics101 scene.



Add a second Link Editor window to show second sphere's PhysObject

- On the Default aspect of the copied sphere's PhysObject, change the Mass value of the Sphere, 1 to 10000 and also the Elasticity to 100.000. Now when you run the simulation, you see the difference by watching the original sphere and the edited values of the second sphere at play.
    - As long as you leave the values on the original sphere alone, you should be able to make changes to the second sphere and run the simulation. If you wish to restore the values, you have the first sphere to use as a reference.
- Once you have explored the Default aspect of the PhysObject, carry on to the next aspect, which is Velocity.
    - Change the Speed values for x … then for y …. then for z on the second sphere.
    - Change the Rotation values for x … then for y … then for z on second sphere.
    - By default all these values on the Velocity aspect are 0.00, so you may wish to experiment with editing values on the first sphere as well. After experimenting a little, you should be able to accurately control the spheres in space.
- Moving along to the next aspect of the PhysObject; Acceleration, follow the same process of editing the second sphere and running the simulation.
    - Again the values for Acceleration default as 0.000, so you should be able to edit both spheres, test the scenario and return the values to default as desired. As you experiment you begin to gain control and knowledge of how the settings function.
    - Once you have a feel for acceleration for speed and rotation, you can begin to flip back to the Velocity aspect and edit values for speed and rotation as well as for acceleration. As you can see, the whole scenario can become quite interesting when you combine different values on different aspects of each object's PhysObject object.
- Next aspect to explore is the Forces aspect. Once again, if you remember the default values for the PhysObject, you can begin to experiment with the two spheres' attributes on the Forces aspect.
    - In physics, opposites attract. Set one sphere to have electrostatic on with a value of 10.000. Set the second sphere to electrostatic on with value of -10.000. Run the simulation and examine the results.
    - Examine the rest of the attributes on the Forces aspect for each sphere.
        - Create a few more copies of the sphere and spread them out in the scene. Adjust their values for Forces attributes and begin to explore the interactivity between the spheres.
        - Set the scene up for zero-gravity and experiment a bit with the Forces attributes. It is a most interesting way to understand trueSpace physics.
- The last aspect to explore is the Constraints aspect. With the knowledge and understanding you have gained up to this point, you should be able to explore the attributes on the Constraints aspect for single sphere, for two spheres and of course for multiple spheres. Keep the scenario simple, change one attribute at a time until you get a feel for what will happen.
    - Add changes to the PhysSpace and PhysEngine object along with constraint attribute changes on the sphere(s). Mix values on all the aspects and see what you can come up with.
    - By the time you finish exploring the PhysObject object and all its aspects and attributes, you

should have a solid foundation of understanding on the subject of physics simulation in trueSpace. As a final exploration of trueSpace physics, find the scene named Gravity in the Physics101 library area. Load this scene and run the simulation. What you should see is gravity at work without using PhysSpace gravity in scene. Check the values for the Mass of the objects within the scene. In short, objects in trueSpace with very large Mass values, have gravity associated with them.

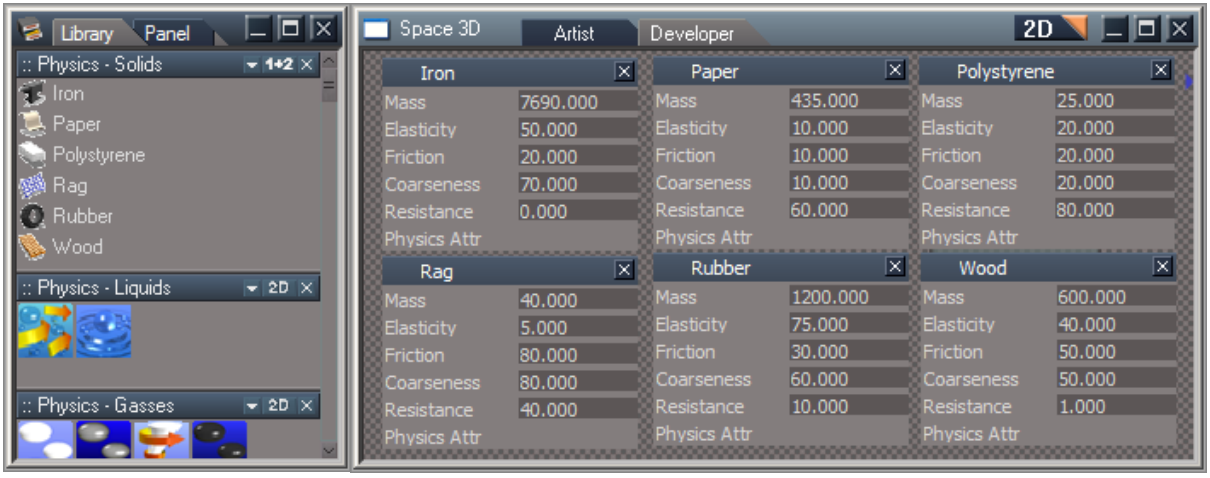


Large Mass objects have gravitation in trueSpace

10.3.2 Tutorial: Using the Physics Materials basic Solids Library

At this point in time, we have covered Physics Libraries and the various tools, which make up Physics in trueSpace. We should now go over the parameters that are having an effect on Physics in trueSpace libraries and tools. At a very basic level we only have a few objects which need to be overviewed. In the illustration below, the same basic attributes exist on each of the Solid materials. If you examine the default values for each Solid, you will see where the attributes are adjusted to emulate these basic Solid materials.

What better way to understand this principle than to introduce a tutorial which shows these basic Solids in action.

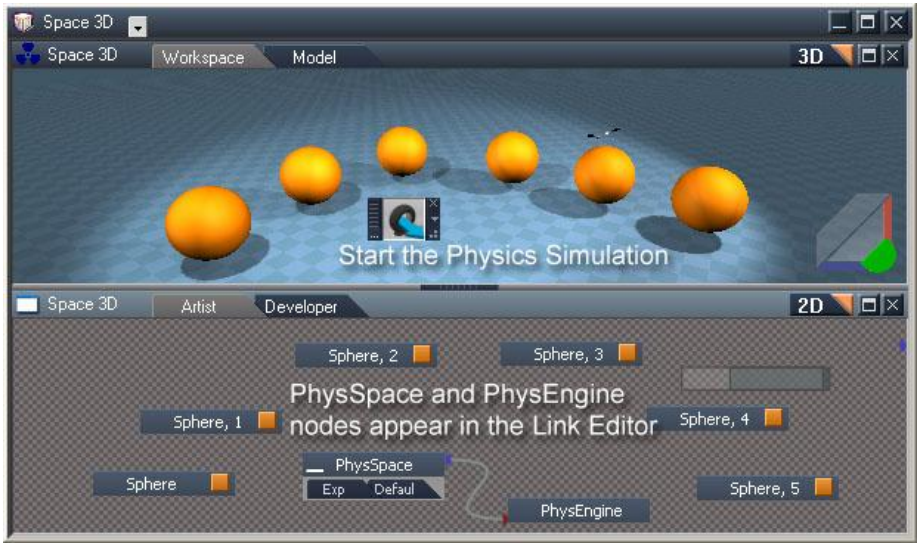


Different values for basic attributes are changed to emulate various Solid materials

- In a new scene, create a single sphere object from the primitives toolbar. Once the sphere is created, move it up along its z-axis so it is approximately 2 units above the ground/grid.
- Copy the sphere 5 times, to give you 6 spheres total. In Link Editor, you can control+drag the initial sphere's titlebar to create a copy. Once you have all six spheres, spread them out as shown below.

Simple scenario to explore the 6 basic Solid materials in trueSpace

- With your spheres spread, select each sphere beginning with the first and drag a material from the Physics – Solids library, onto the selected sphere in the Workspace. When finished, each sphere has a different basic solid material.
- Once each sphere has a solid material the simulation can be started. Note the PhysSpace and PhysEngine objects appear in the Link Editor.



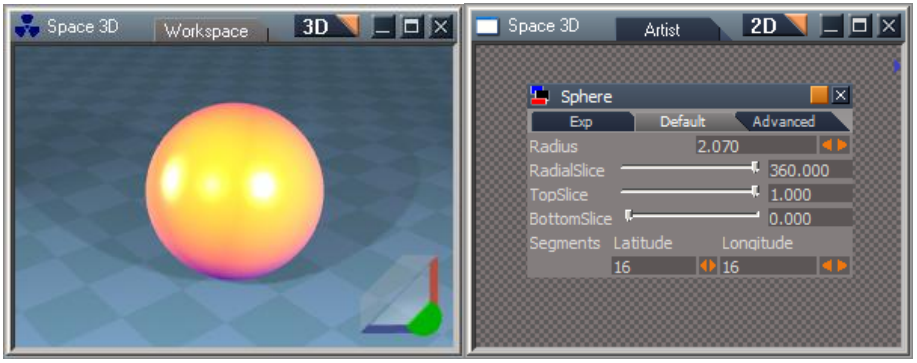
Start the simulation and observe each sphere's physical attributes in real-time

- In the Link Editor, enter one of the spheres and locate the solid material object. It will have the default settings for the physical attributes for that sphere. You may begin to change values if you wish. To see how the changes you make influence the physics simulation, simply re-start the simulation and observe.

➢ Back to Section

10.3.3 Tutorial: Basic Simulation

1. Load the PaintSphere object from the Base library into the Workspace.

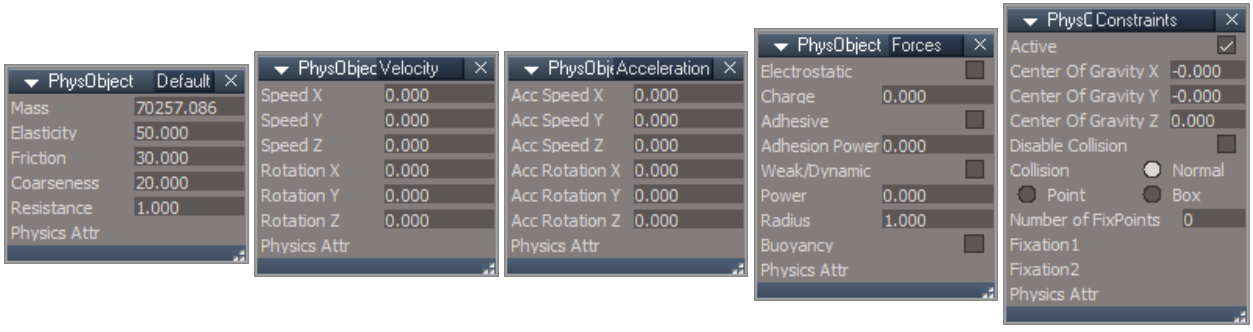


The scene with a sphere loaded and the Link Editor opened

2. With the Sphere selected, click on the icon  to assign physics attributes to the object.
3. To set or modify the physical attributes for the sphere, you can either change the values in the Panel in the stack or enter the object in the LE and set appropriate values for the input attributes of the PhysObj object (or PhysGroupObj if it is a group object).

… and a look inside the sphere after creating Physical Attributes for it

The composite image below shows some of the physical attributes available to you.



PhysObject has 5 specific tabs with attributes unique to that tab

4. For group objects with many sub-objects, you can specify different physical attributes for each sub-object. Simply enter the sub-object's in the LE and set values for the input attributes of PhysObj (currently only elasticity, friction, coarseness and weight are implemented). If you do not wish to set different values of phys attributes for sub-objects, edit only the values for input attributes of the PhysGrObj object.

5. Repeat steps 1, 2, and 3 for the other objects that you wish to include in the simulation.
6. To run the simulation, click on  .
7. If you want to stop the simulation, click on  again. If you do not stop the simulation by clicking on  , the simulation will stop automatically when the simulation time exceeds the value in the input attributes for "Total Time" of the PhysEngine object.

8. The precision of the computation can be specified by modification of the Time Step attribute value of the PhysEngine object. Higher values mean greater precision. The speed of the simulation may decrease, however, with higher values of Time step.

10.4 Characters and Physics

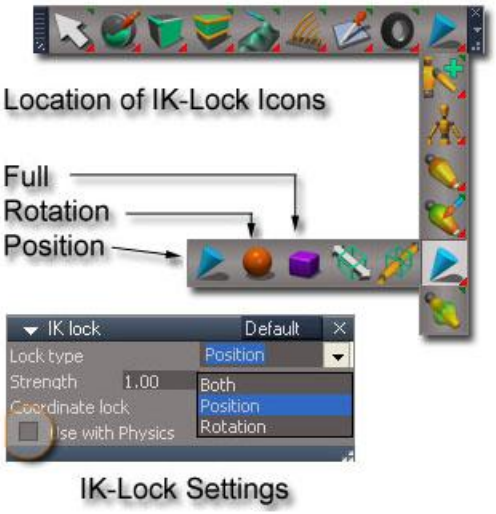
Workspace physics allows the assignment of physical attributes to characters and generation of character animation using physics-engine. To assign a physical attribute to a character, first you select the character object in Workspace or in the Link Editor, then press the Add Phys Attr button  from main toolbar. The character object must have a connector "Skeleton" exported to be recognized as a valid object for character-physics. This connector is located on the skeleton object panel that defines skeleton for character.

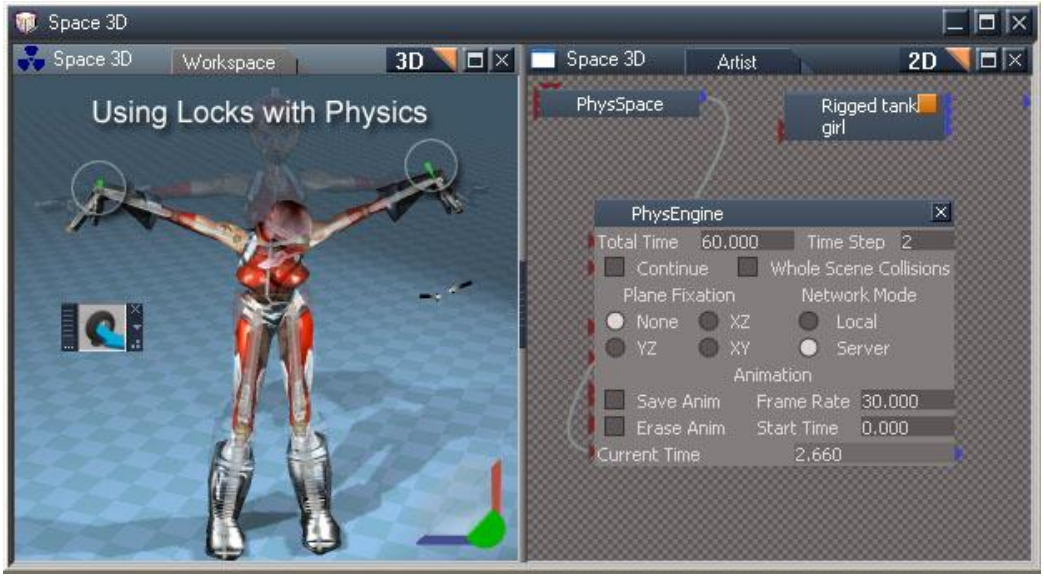
Assigning Physical Attributes to the character

Character must contain a proper Skeleton object

To start a simulation with characters press button from main toolbar  .

Running simulation with character –unconstrained

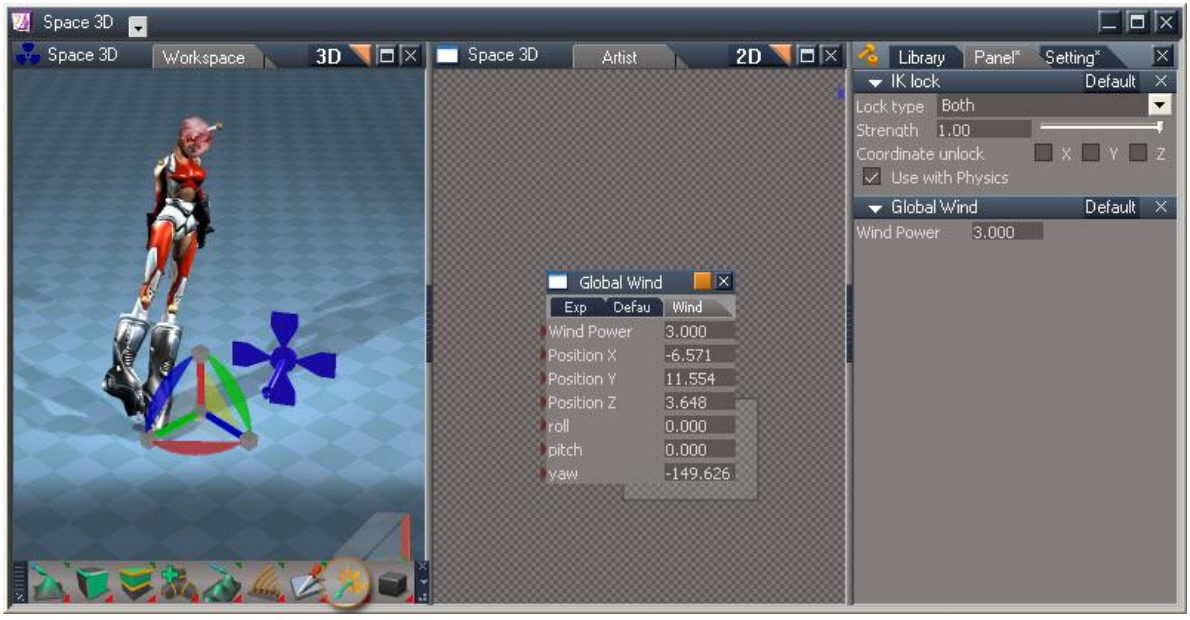




Simulation using two locks on character's wrist joints

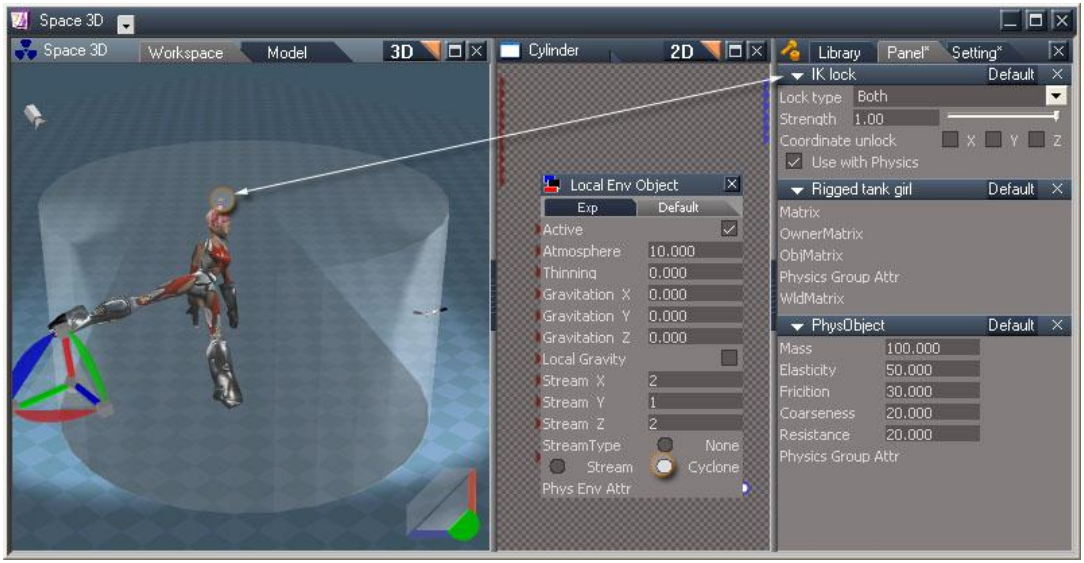
The character can be constrained during simulation by using several locks. You can add a lock before the start of a simulation using the character editor toolbar; the properties of locks are the same as for inverse kinematics (there are three types – rotational, position and full lock). For the moment, all added locks are active and have effect on the character simulation. To remove or disable lock from the character object, you must delete it.
The character may also react with the Global or Local wind objects.

Set up the character for use with physics environment



Add and edit Global wind to affect the character

With the Global wind selected, activate physics and use the navigation controller to change the direction of the wind. You may also wish to experiment with surrounding the character with for instance a cylinder and setting the cylinder to be a Local Environment with a cyclonic attribute for the Stream.



Using Local Environment with character

By combining various forms of physics, you are able to create very diversified scenarios for characters to react in.
In addition to using the Add Physics Attrib button from the main toolbar, physics can be assigned to a Character or a Skeleton from the Physics-Solid library of predefined physical materials. Double-left-click on given material's icon or by Drag and Drop of the material on the Character or Skeleton object. You can enter individual objects in the character's hierarchy, to edit each as desired.

PhysObject displayed in several locations in the hierarchy of the character

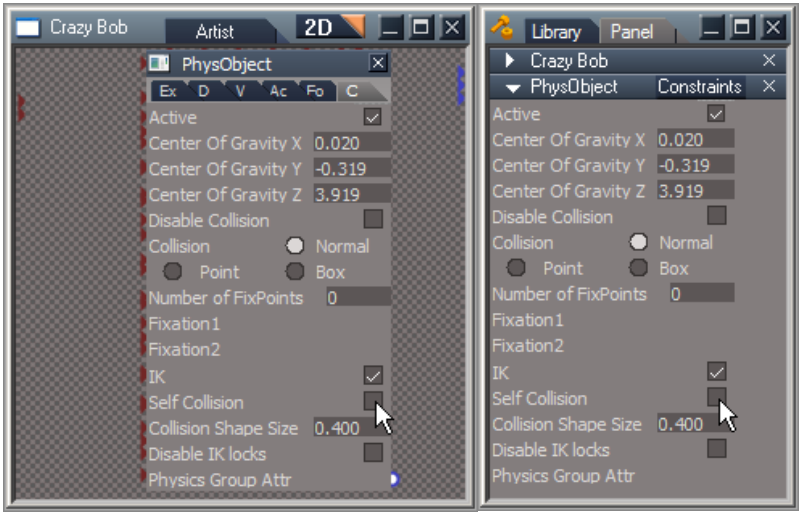
Physical attributes are assigned to a character similar to the hierarchical (group) object. You can modify these attributes for whole object (see *PhysObject* object panel), by modifying single attributes of the *PhysObject* object. The *PhysObject* object that controls the physical parameters of a skeleton object has one output attribute: *Physics Group Attr.*

It is possible to set individual physical parameters for each single bone of a skeleton. After assignment of physics to a skeleton, each bone object has one exported output attribute named *Physics Attr* where all the bone's physical parameters are stored. This connector is exported from *PhysObject* object that you can find in the bone encapsulator. After physics is added to a skeleton, the default parameters for each bone are the same as global skeleton physical parameters – all bones have the same values of elasticity, friction, resistance and so on. However you have the ability to set parameters for bone(s) that can be different from global setting, by editing values of attributes on bone's *PhysObject* panel– for example you can set different elasticity for hands and legs of a character object which will result in different behaviors during collisions with the environment.

PhysObject top level for the group controls the PhysicsGroupAttr as shown by its output connecter and controls the entire character. Looking inside Crazy Bob's skeleton and looking inside a bone shows the Physics Attr output connecter, the PhysObject has been changed to rubber; this physObject only applies to the individual element within the hierachy.

Bone 2 has Rubber material applied

Four main attributes control or determine the behavior of character object in simulation; they can be found in Constraints aspect of Character's *PhysGroupObject* panel. By default, characters collide with all objects in the scene and collision between character's parts are ignored (bones could go through each other). To enable collision between parts of skeleton, you need to check out attribute with name *Self Collision* on the *PhysObject* object panel. You can turn on or off self-collisions during running simulation.



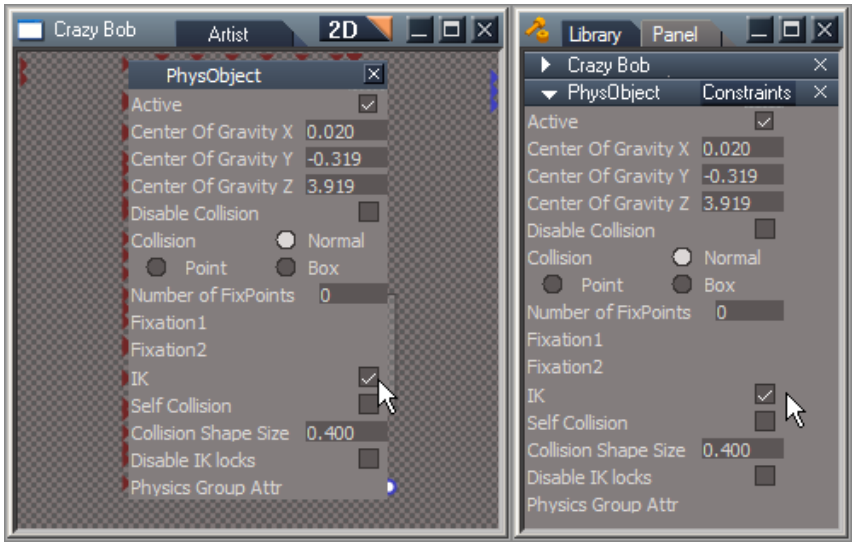
Enable Self Collision from Constraints aspect

Collisions between bones and other objects in the scene are checked using bone collision shapes (they are similar to bounding boxes) that envelop character's bones and not by use of skinned mesh. With attribute Collision Shape Size

you can modify and set the thickness of these shapes. Default value of a shape size is 0.4, for example if you increase the thickness of bone's shape, collision will be detected further away from the bone. You can interactively change this attribute during simulation.

When you edit a pose of a character for example with Dynamic Pose tool, you can use many IK locks (rotation, full, position). To enable these locks in the simulation, check the Disable IK locks attribute. By default, all locks are disabled. Enable or disable of locks can be done interactively during the simulation (toggle on/off).

The IK attribute allows you to turn off a character's physics for object, if this attribute is not checked; object will behave like any ordinary solid group object without considering its joints and bones. By default, if you assign physical attributes to a character object, this option is enabled. You should set whether object will have character-like or solid object-like behavior before running a simulation, it can not be changed during simulation.

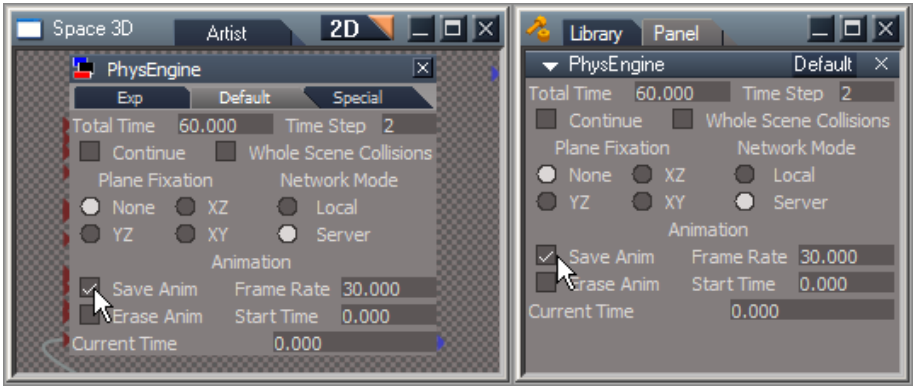


attributes in PhysObject panel for character object with physics (Constraints aspect)

10.4.1 Generating key frames using physical engine

You can save the movement of objects that are animated using physical simulation to key frames and later you can replay the result of simulation without the need to rerun physics. Later, you can use the generated key frames for more complicated and more complex animations using Workspace animation subsystem with help of clips and tracks.

- Reference : Chapter 9 Animation



Saving physical animation into key frames

To save the result of the simulation to key frames, open the default aspect of the PhysEngine object panel (object is located by default in Space 3D encapsulator in the Link Editor) and check the attribute *Save Anim.* After starting the simulation, movement of objects are saved into key-frames at the rate specified by the value of the *Frame Rate* attribute. For example if you set this value to 10, ten key-frames per second will be generated for every object included in simulation. When total time of simulation is for example 60s, 601 key-frames will be saved (including 0. frame). The value of the *Frame Rate* determines the quality of saved key-framed animation. If this value is too low, the playback of the animation may be different from the actual simulation (live), since the positions between key-frames are calculated using interpolation based on neighbor key-frames. The interpolated position of the object can be different in comparison with position in simulation at the same time. This will be visible mainly in a situation where there are many collisions and movement is fairly fast. Optimal value of *Frame Rate* is between 10 frames per second and 25 frames per second.

In addition to *Frame Rate*, you can specify the number of the first generated key-frame by value of the *Start Time* attribute. For example, value of 5 means that the first key-frame after start of simulation is saved at frame 5. For example, a 10 second long simulation would key-frame from frame 5 to Frame 316, with key-frames set at 30 fps frame rate.

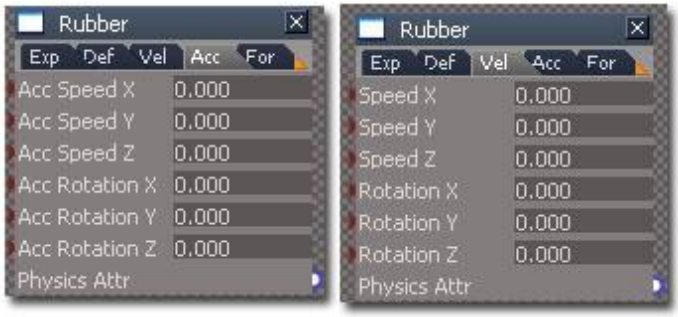
You can toggle saving of key-frame on and off during the simulation. With simulation running, you can start or stop the generation of key-frames at any time, by toggle of the checkbox for *Save Anim*. You can also adjust the *Frame Rate* attribute when simulation is running by changing the value during the running simulation. The simulation will use new values as you adjust them.

10.4.2 Setting speed and acceleration for character object with physics

When you assign physical attributes to a skeleton and then start a simulation, by default all speeds are zero and

skeleton starts to move in gravitation field down towards the ground. To change or to set initial speeds of skeleton you will need to set initial speeds using speed tools (widgets) or by editing of values of attributes on the *PhysObject* object with output attributes *Physics Group Attr*.

The Vel and Acc aspects of physics panel contain the required settings. There are two possibilities to set/change the speed - either for entire skeleton and/or for single bone within a skeleton. To set speed for whole attributes you need to change speed attributes value of skeleton's *PhysGroupObject*, to set speed for an individual bone, edit the bone's *PhysObject*. Speed and acceleration of bones are always relative to skeleton object, final bone's speed is sum of global skeleton speed and bone speed from panel. You may change speeds using LE panels during simulation – new values will be applied immediately to objects. For more advanced uses it is possible to write scripts to adjust these values if desired.



Velocity and Acceleration aspects of PhysObject panel

Speed and Rotation tools

There are four tools for modifying of initial kinematics attributes of characters with physics and are used as described earlier:

-  Speed  Rotation  Speed Acceleration  Rotation Acceleration

All speed tools can be used in two modes – in the first mode, user can set speeds for each bone individually; second mode sets global speeds for whole skeleton (character).

Buttons for speed tools

To start speed tool for character object with physics, first select the character. Then you can apply speed tools.
The first mode (speed for individual bones) is started by left-click on the tool. You will notice x-ray view mode for character is turned on and you can pick which bone initial speeds should be set for.
After clicking on a bone, the speed widget for that bone appears. By dragging the widget, you change the initial speeds for that bone. To set speeds for other bones, pick and repeat the widget action again for it.
To exit the tools, R-click anywhere in the Workspace 3D window.

Setting of Speed and Acceleration for bones

Setting of Rotation and Acceleration Rotation for bones

To set speeds for whole character, press ALT key together with L-Click on button of tool you wish to use. The speed widget should then appear on the skeleton and you can use it for changing initial speeds of character. To exit speed tool, again R-Click somewhere in Workspace.

You can combine setting of initial speed with widget and use of phys object panel. When you activate widgets for bones or for whole character, R-Click on button. Phys object panel for selected bone or character will open in Panels Stack – you have to switch manually to Panel Stack if it is not visible. Using Velocity or Acceleration aspect you can exactly set or modify the size and direction of initial parameters for selected object. To see speed widget for selected object while editing values of panel attributes (after panel is opened, widget tool is deactivated – it is limitation of current implementation), simply activate speed tool again by L-Click (or combination ALT+L-Click if you work with whole character) on tool's button. Then, you can modify speeds with widgets and/or by setting values in phys object panel – if you set values of speed attributes in panel, you will see change of widget orientation and vice versa.

Use Alt Key when setting Speeds or Rotations for whole character

The other way to interact with skeleton during simulation and modify its speeds is to use The Phys move tool  .

Activate this tool by L-Click on phys move tool icon and then L-Click on an arbitrary part of the skeleton. You will be able to change the speed of that part and thus you will be able to drag the skeleton in any direction you wish. The Phys move tool is available for use only when a simulation has started and the physics engine is at work.

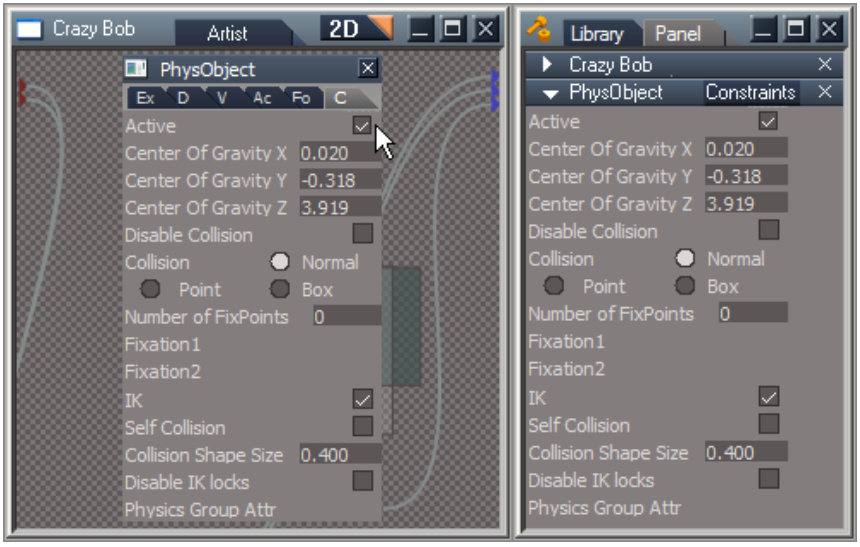
10.4.3 Enabling interaction between characters and objects with animation

Enabling interaction between characters and objects with animation and objects and characters with physical attributes in simulation.
In simulation, the physics engine calculates all the movements of characters and objects with physical properties. If characters/objects have animation recorded beforehand, this animation is ignored (for objects/characters with phys properties) and new animation is calculated by the physics engine.

There is however the possibility to incorporate into the simulation, characters and objects with pre-animated movements – in this case, animation-system will manage the movement of characters and objects, according to saved key-frame information. The physics engine will calculate the interaction between these types of characters and other objects with physical attributes. These characters and objects are inactive (from a keyframe point of view) in physics engine.

To enable such behavior, load a character or object with pre-recorded animation. If it has no physical attributes, assign them as desired. Open PhysObject panel for the character, go to Constraint aspect of the PhysObject, and un-check Active. The Active Attribute specifies whether the given object is to be included in the simulation. If Active

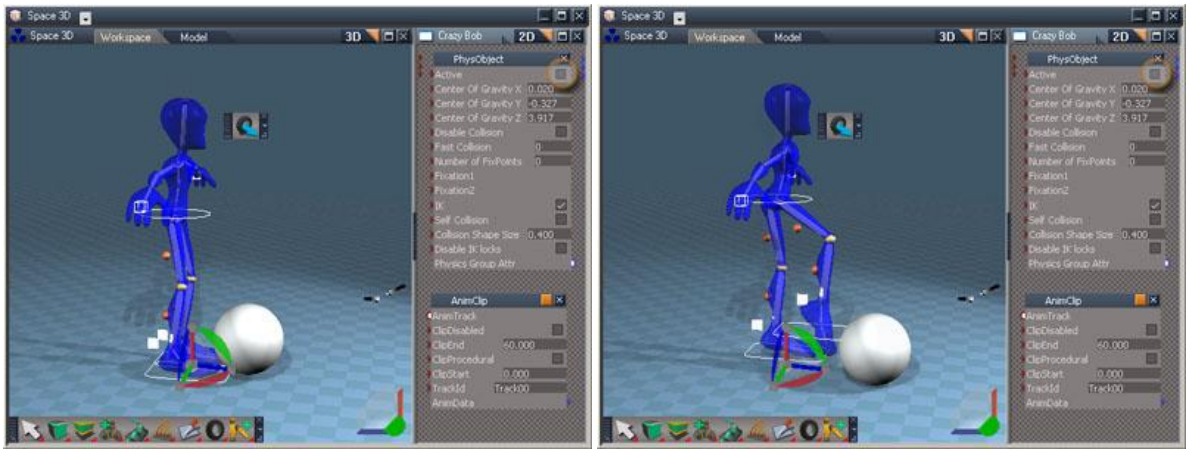
is not checked, the object will be inactive in the simulation and will behave as a static object. For character physics, if the character is set to inactive in simulation and has animation recorded, movements of the character are included in the simulation and collisions with other object are monitored. This allows interaction of the animated character with other objects in the simulation.



Constraint aspect of phys object panel

10.4.4 Soccer Bob Tutorial

Applying this principle of animated characters in physics simulation, the illustrations below demonstrate the Crazy Bob character and a soccerball object in a scene. The tutorial is easy enough to set up. Bring in the CrazyBob character and record a few keyframes at frame 30 and frame 60, moving his right leg as shown in the images below. Bring in a ball/sphere object and assign physical properties to it. Place it in the path of the leg-movement you created. Lastly, select the character, assign physical properties to it and enter the character in Link Editor. On the PhysObject's Constraints tab, un-check Active attribute. Once the simulation is started, the physics engine calculates the speed and force of the leg/foot as it collides with the ball.



Using keyframed animation and physics in trueSpace

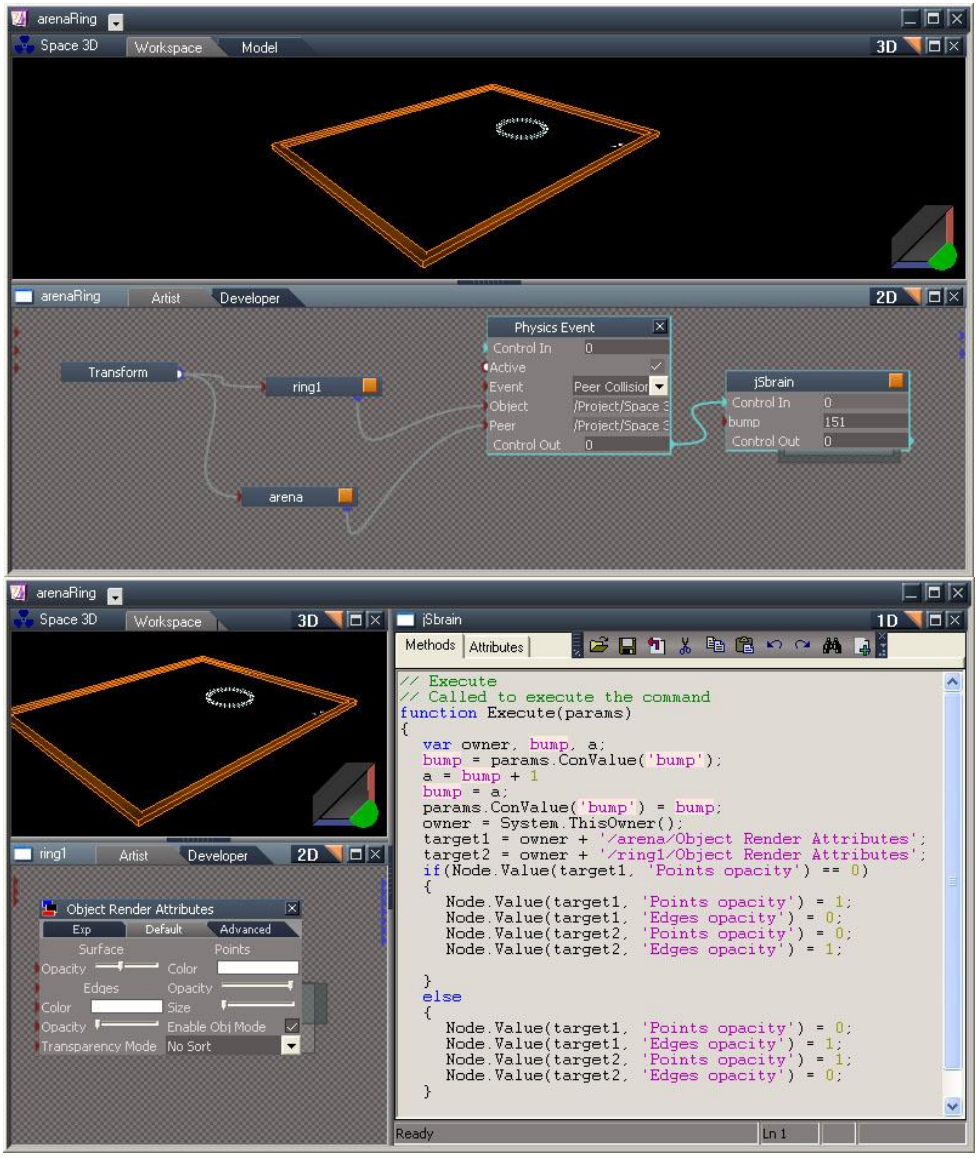
There are any number of different scenarios you will think of to try this type of scenario in trueSpace. The possibilities are most interesting.

- Reference : Chapter 9 Animation

10.5 Scripting the Physics Event object

Various events to detect

There are a number of different physical events, which the Physics Event monitors. You specify which event you wish the Physics Event to monitor. When that event occurs, the Physics Event fires a control-pulse, which would then trigger in our example, a script designed to toggle visibility of vertices and edges.



Physics Event object used to detect Peer Collision

There are also exposed attributes of the Physics Event, which allow you to access the information via scripting in trueSpace.

Check the IRiPhysicsCommand Struct Reference for specific information on what is exposed and how to access this data.

- Reference: Chapter 11 Activities