Creating joints for the NovodeX MAX exporter

(a step-by-step tutorial by Pierre Terdiman)
p.terdiman@wanadoo.fr
Version 0.3

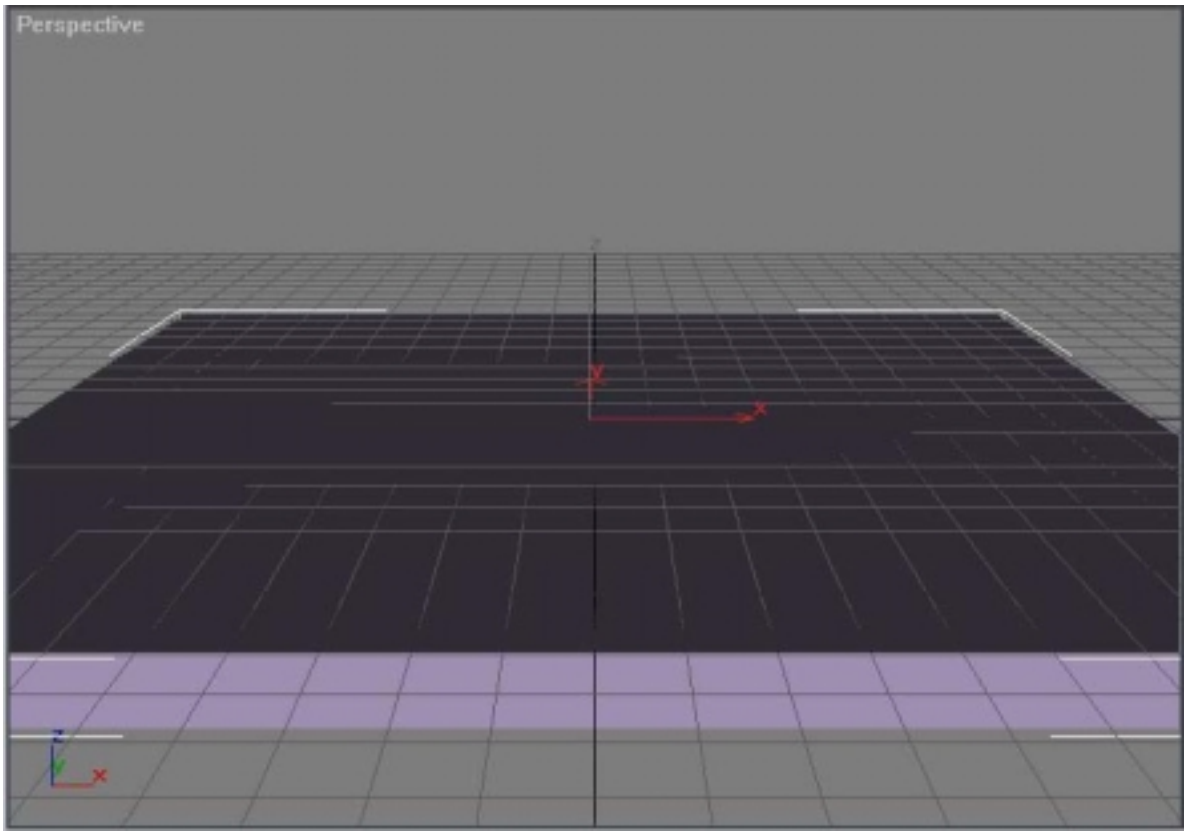
I) Creating a hinge

Here we'll see how to create a *hinge* joint compatible with the *NovodeX* exporter.

Step 1 : create a ground plane

We first create a ground plane, where our meshes will lie. This is not mandatory, but it's prettier than the built-in infinite plane in *NovodeX*' viewer. We'll also use this as an introduction to *User Properties*.

So, create a ground plane using, for example, a box. It should look like this :

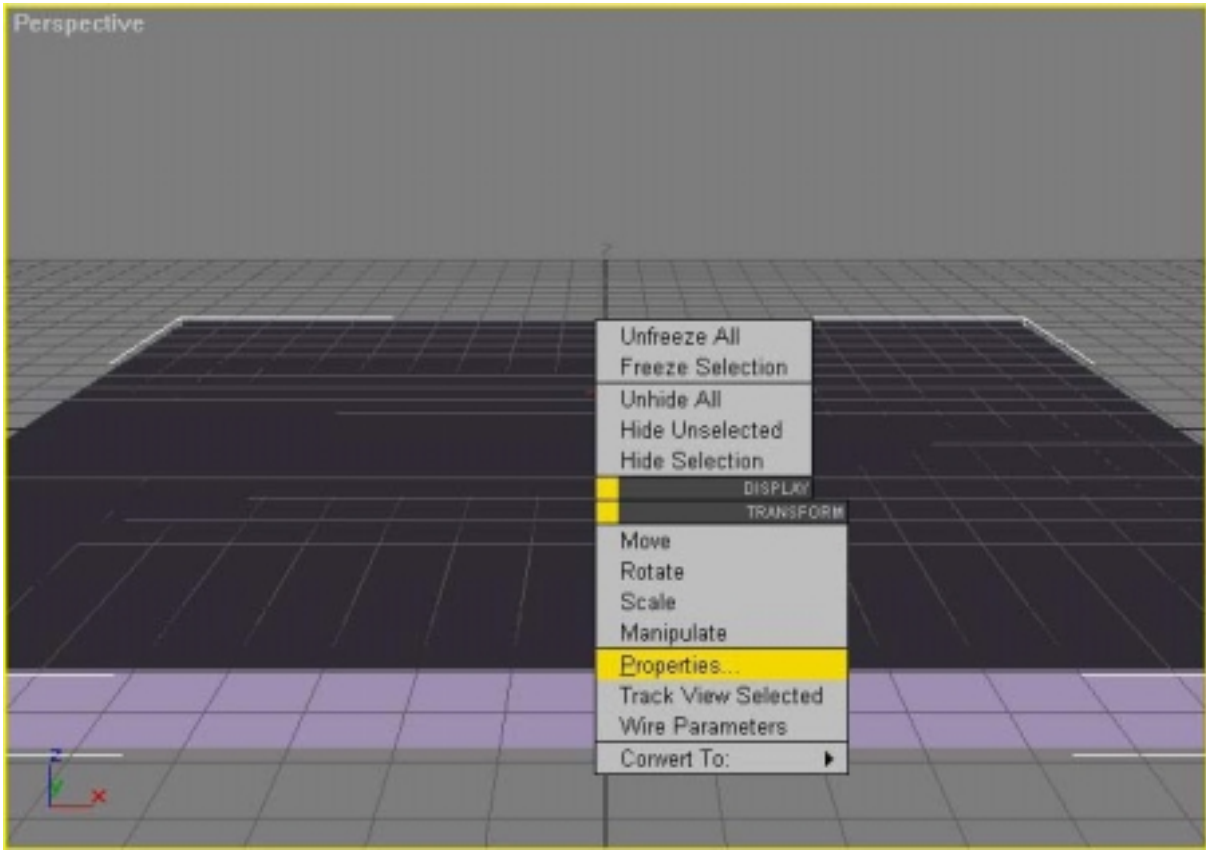


If you stop right away and export this scene without further ado, you'll end up with a physically-controlled box. The box will be subject to forces (e.g. gravity), and you'll be able to drag it with the mouse.

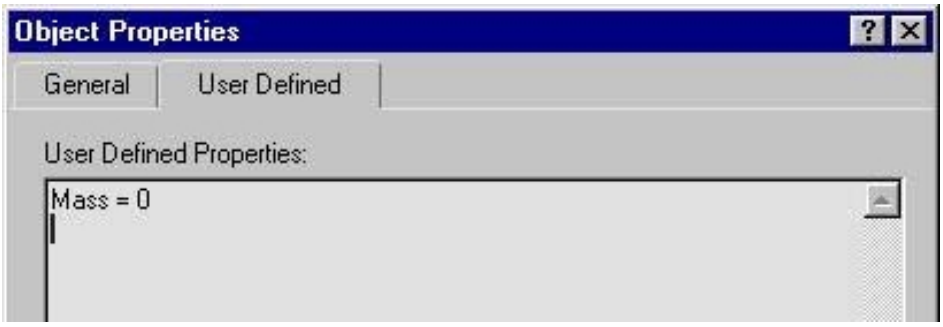
This is not what we want. We want it to be a fixed box.

So we'll mark it as such, and we'll do that by setting its mass to 0. This is a convention used in most dynamics simulators : objects whose mass is null don't move.

Since MAX doesn't know about masses, we'll take advantage of User Properties here. Select the box, press the right mouse button and select **Properties...** as in the picture below.



Then select the **User Defined** property sheet in the **Object Properties** window, and define the mass that way :

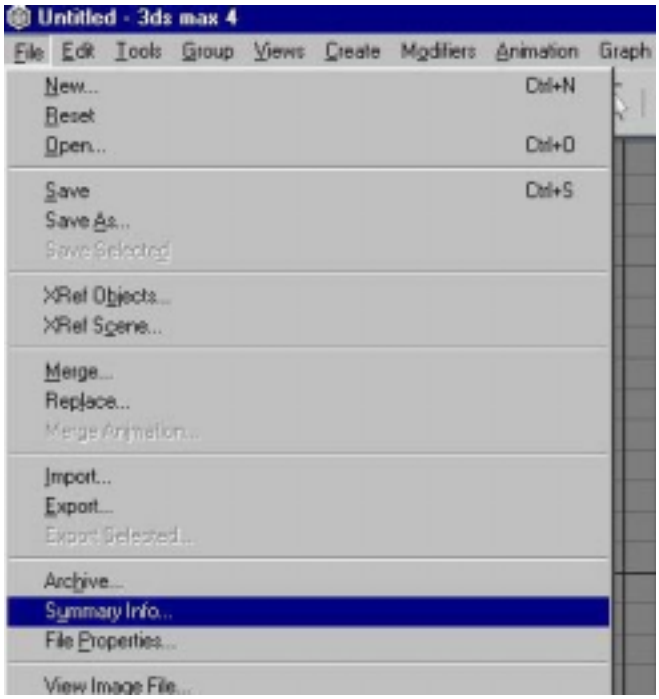


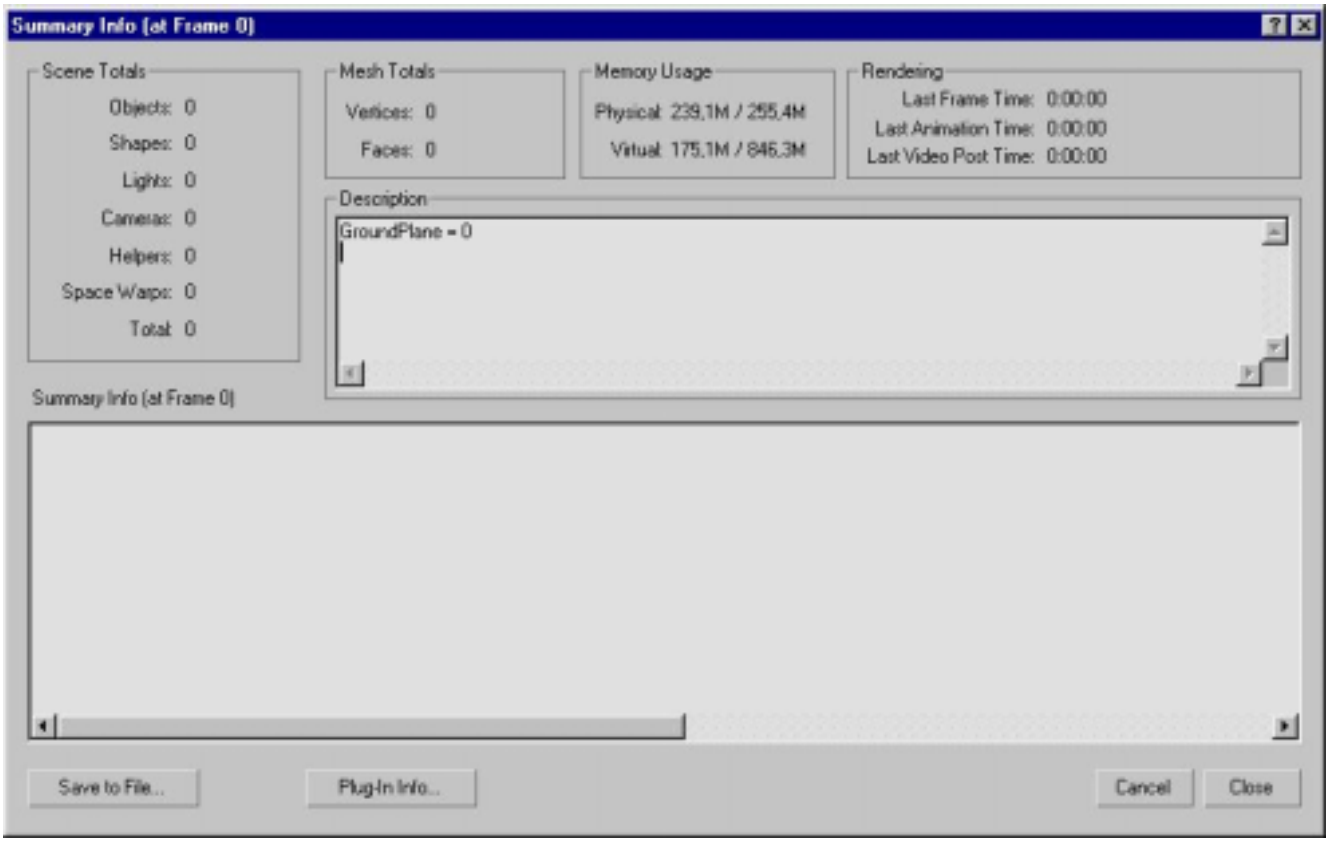
You could also write "Mass 0" (without the *equal* sign) or "mass 0" (detection is not case sensitive).

Now, export again. The box doesn't move anymore. It's now our ground plane.

The *NovodeX* viewer contains a built-in, infinite, invisible ground plane. We don't want it to conflict with our own ground plane. There are two ways to do that :

- create your box so that it doesn't cross the default MAX grid
- or disable the viewer's plane. You can do that by writing "GroundPlane = 0" in the scene's User Properties. Since the scene itself has no node we could select as we did before, we use the scene **Description** instead. In MAX, you can access it by clicking **Summary Info** in the **File** menu.





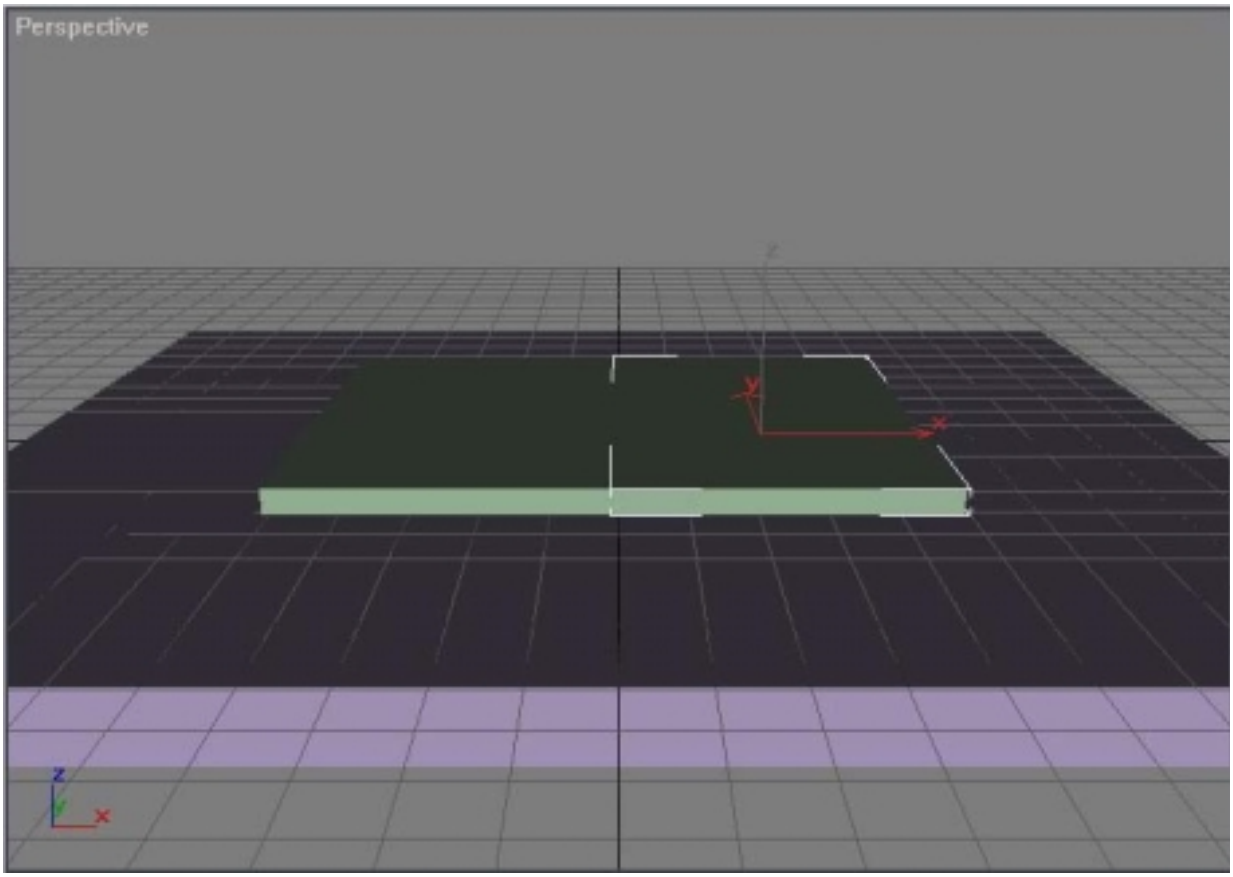
Note that our box is made to look like a plane, but it's not. Collision detection is more efficient with a real plane, but for this tutorial we don't care.

Well, now that we covered the basics, let's move to our actual goal : a hinge.

Step 2 : create meshes

Our hinge will link together two meshes. Before creating the hinge, we need to create the meshes. We'll use two new boxes, scaled so that they look like plates. You can make one an instance of the other (it's more efficient during the simulation).

Create two plates lying on our ground plane, move them so that they just touch, like in the image below.

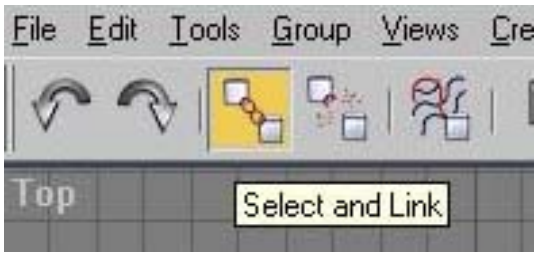


Since we want them to move, we don't set their masses to 0 as we did with the plane. However, we could explicitely define their (positive) masses using the same way. For now, we'll just let the engine use default mass values.

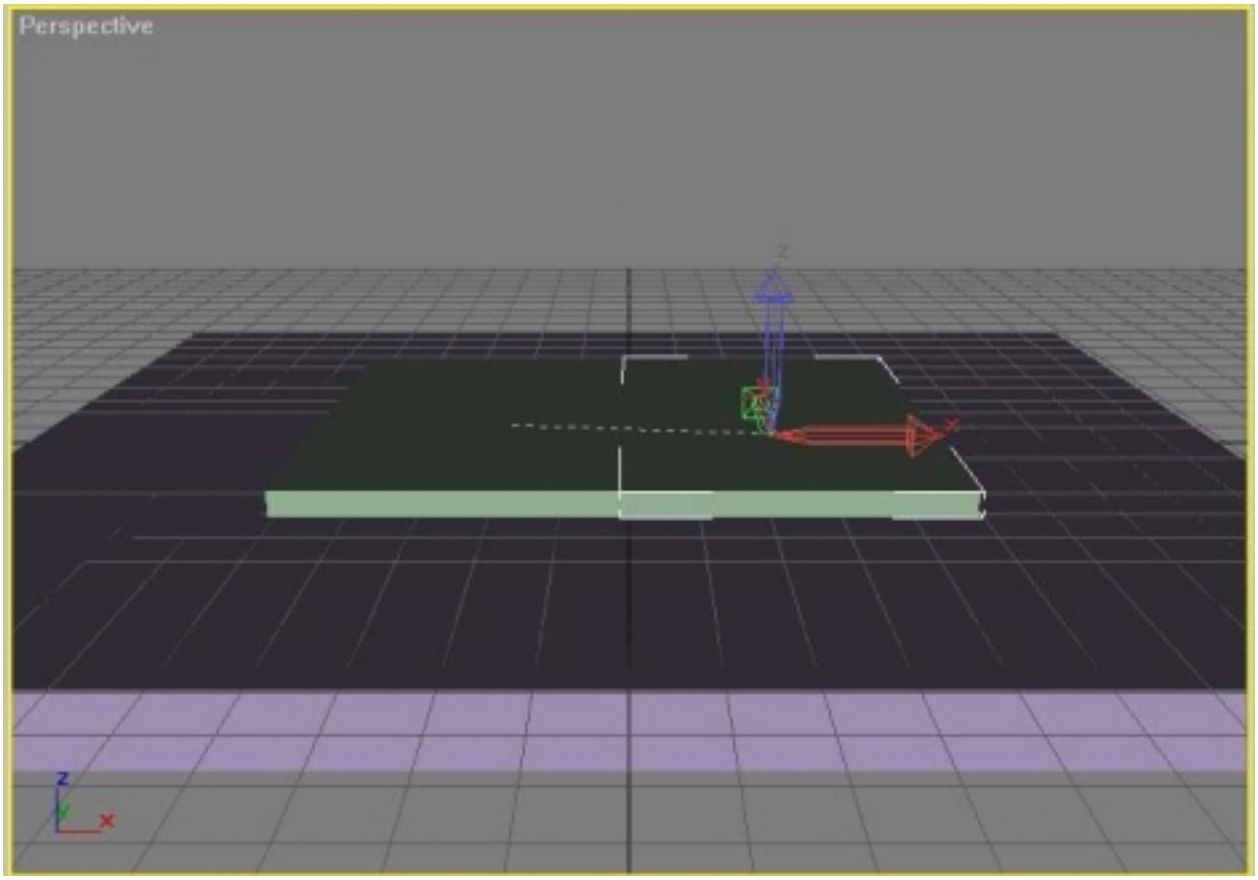
If you export this scene now, you should get two physically-driven unconnected boxes, colliding with the ground and reacting in a realistic manner.

Step 3 : linking the meshes

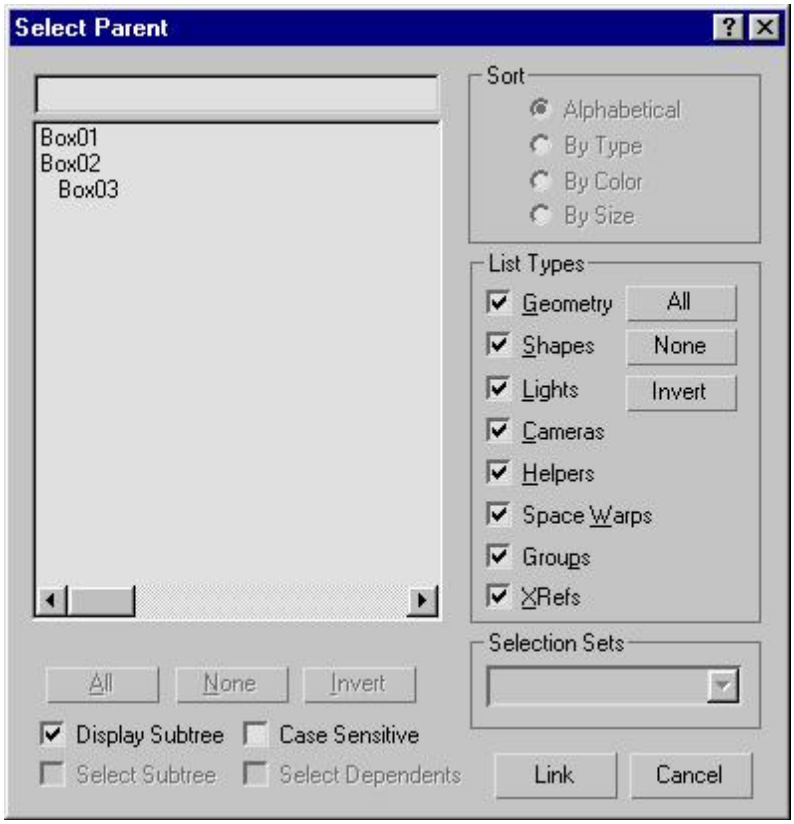
First, we need to link the meshes in MAX. We'll do that using a standard parent-child relationship. To setup one of the mesh as the child of the other, click on the **SelectAndLink** button.



Then click on the child mesh, keep the mouse button pressed, and drag the mouse pointer over the parent. You should see a dotted line starting from the child's pivot point, ending on current mouse position.



Release mouse button to create the link. You can check the link has been correctly created by displaying the MAX internal hierarchy. To do that, click on the **SelectByName** button (see image below). This window pops up :

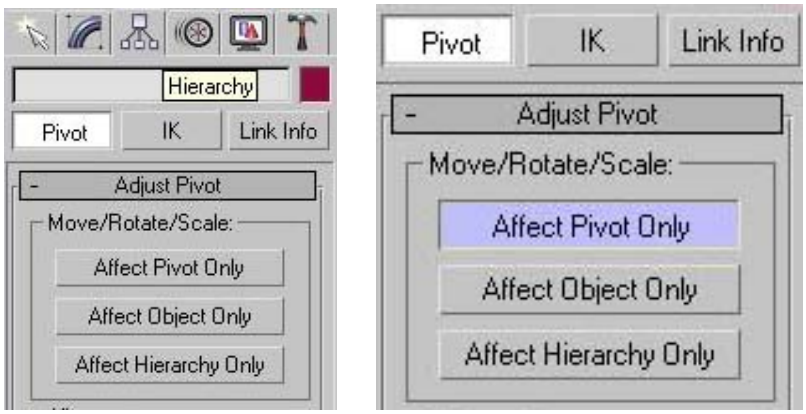


Then click on **Display Subtree**. In our example, *Box3* is now a child of *Box2*.

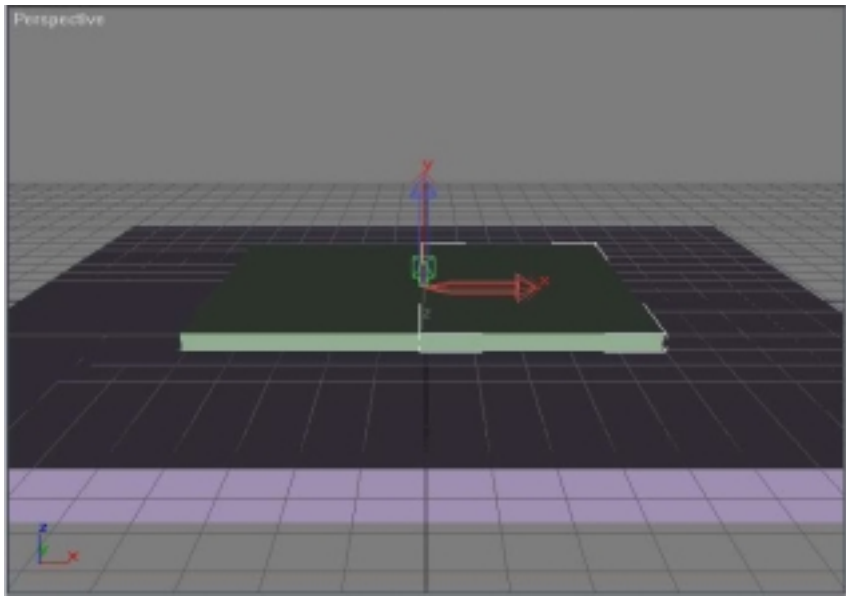
Step 4 : placing pivots

Pivots points play a very important role, regarding joints. To begin with, and as the name suggests, the pivot point is used as center of rotation. So the pivot's position is important and should be set carefully.

To display pivot points in MAX, select the **Hierarchy** panel on the right, then click the **Pivot** button. To further move the pivot, select **Affect Pivot Only**.



That way you won't move your object accidentally. You'll only be able to move the pivot point. Translate it so that it lies between the two linked boxes, as in the image below.

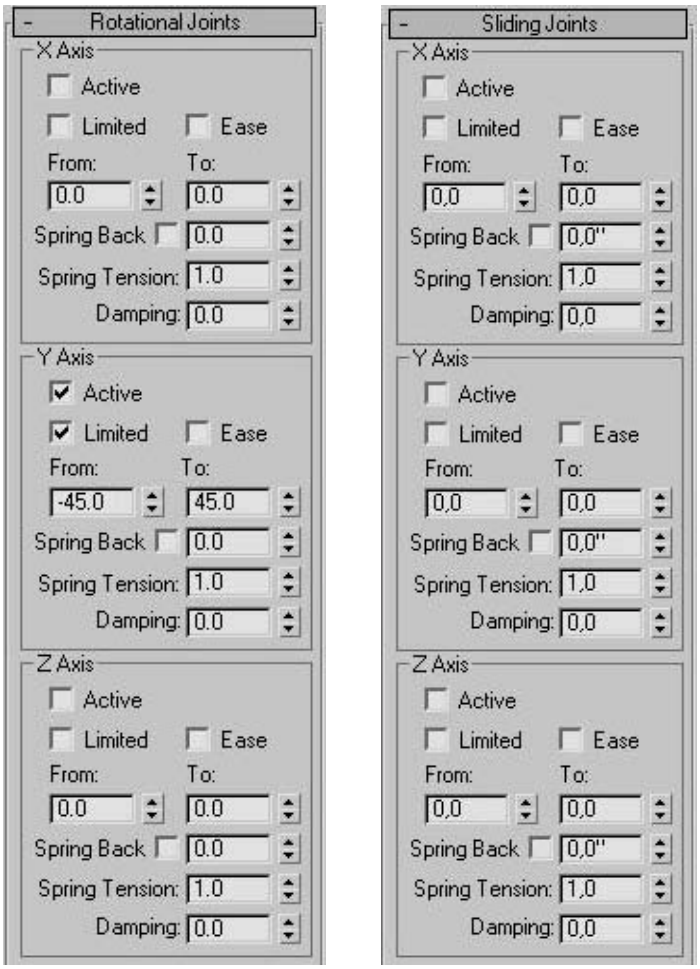


Now, a word about pivot's rotation. You should make sure both pivots (for the parent and for the child) are aligned with the objects' local axes. This is not mandatory, but it solves some internal MAX issues (mainly gimbal locks on IK axes). In short, this is safer and you should do that for the exporter to work well.

Aligning the pivot's axes can be done automatically in MAX by clicking on **Align to Object**.

Step 5 : setting joint parameters up

Now, we're ready to define joints parameters. In the same way you selected the **Pivot** panel, now select the **IK** one. Again, we're working here with the child mesh. On the right appears a big panel with lots of information. Don't panic, we'll only use a limited set of parameters here. Modify the **Rotational Joints** so that you get those settings :



We want to create a hinge, which is a joint with a single rotational *DOF* (Degree Of Freedom). That's why we only set a single axis as active (the **Y Axis** in our example). We assume all **Sliding Joints** are inactive (as in the image above), since this is the default MAX state for them.

Now, why the **Y Axis** ? It has to do with the object's pivot. It's not always clear, even in MAX, around what axes the object is going to rotate - at least from a theoretical point of view. So just modify the **From** and **To** sliders to directly check the results in the viewport ! As long as your pivots are aligned with the objects, you should be fine. In our example, we used the **Y Axis** so that the linked plates rotate as if they were, say a book cover.

Next, setup limits as on the image above. If the **Limited** checkbox is on, **From** and **To** fields will be exported, and the engine will make sure limits are respected during

the simulation. Else, if that checkbox is off, the objects will rotate around the hinge freely.

The **From** and **To** limits should be set visually, that's the easiest way. Just move the sliders and check the limit positions within the viewport.

That's it !

You created your first hinge. Export the scene now, and you should get two physically-driven meshes, linked by a hinge along their line of contact.

What about other IK parameters, you might ask ? Don't bother with them : they're useful in MAX, but as far as the *NovodeX* exporter is concerned, they're *not.*

We went into all little details to make sure everything was clear, so it might have looked a bit difficult. But actually, with a bit of practice, it's quite easy.

If you don't believe me, jump to next chapter for a faster-than-light prismatic joint.

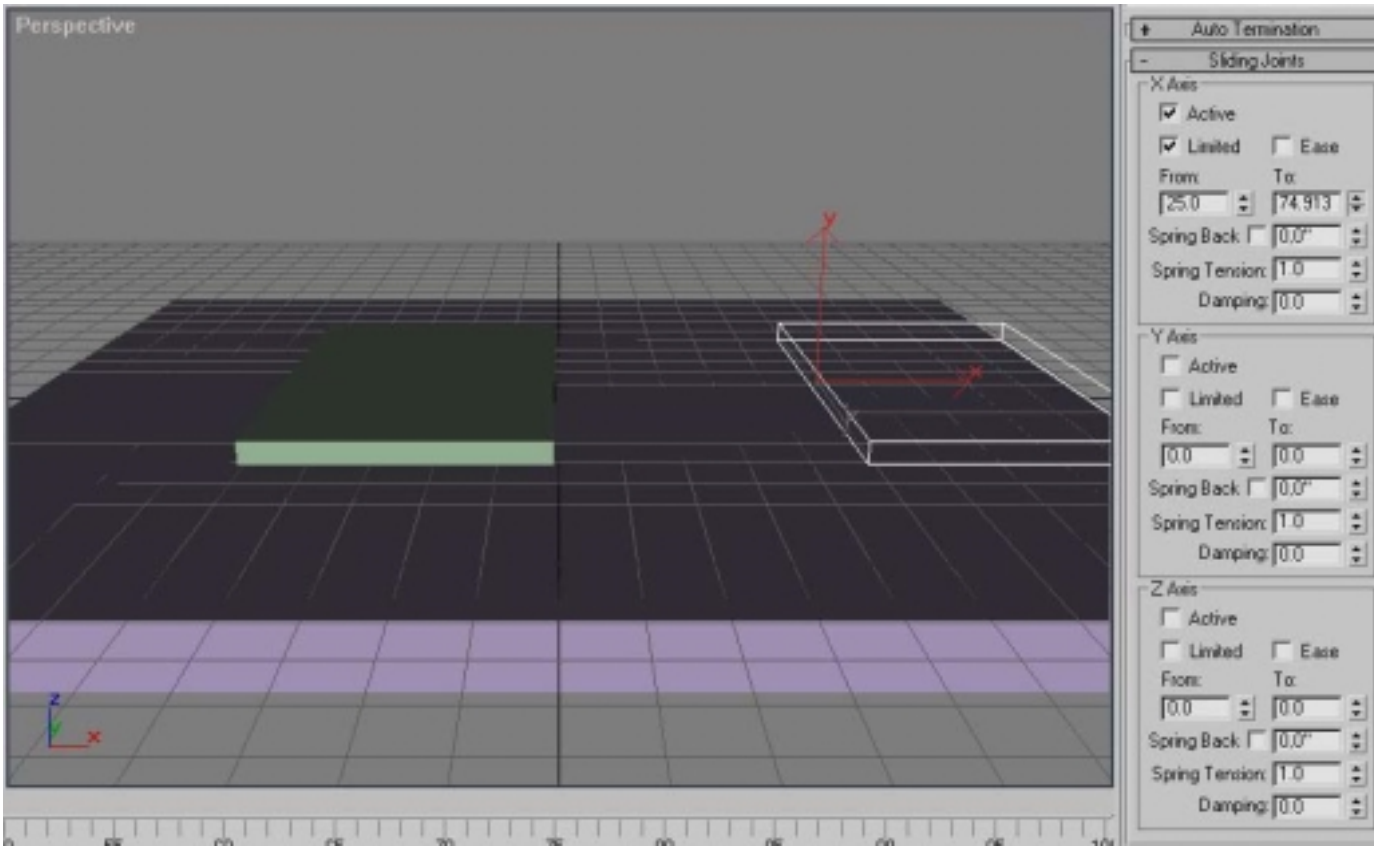
II) Creating a prismatic joint

Armed with our previous hinge experience, we'll now be able to transform our hinge into a *prismatic* joint in no time.

This is very easy :

- disable all **Rotational Joints** (mark them as inactive, i.e. uncheck **Active**)
- setup **Sliding Joints** as in the image below

Note the viewport : when moving the **From** and **To** sliders, ghost objects (their bounding boxes) reflect changes in realtime.



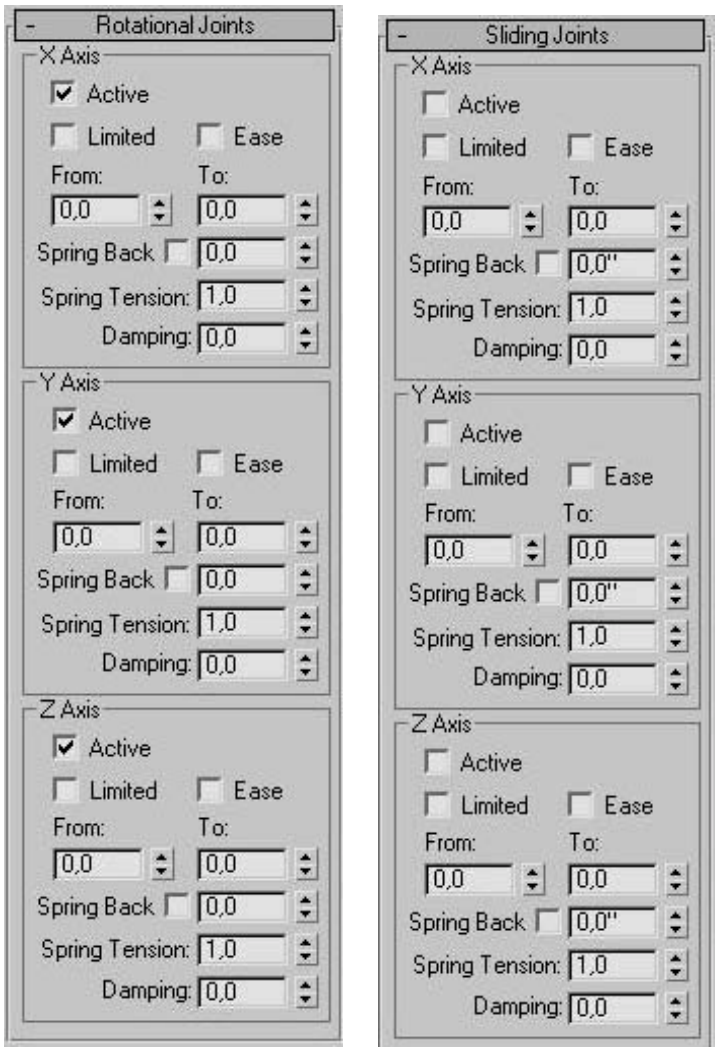
Export again. The hinge is now a prismatic joint ! It would have been just as easy to create a different joint, as you'll see in next sections.

III) Creating a spherical joint

You should be able to create a *spherical* joint yourself by enabling all 3 **Rotational Joints**, and disabling all 3 **Sliding Joints**.

Actually, you'll notice this is the default MAX state when tou enter the IK panel for the first time. This gives birth to a weird little problem : if you just link meshes as explained until now, then export the scene with default IK settings.... no joint gets created, *even if the IK parameters map a spherical joint*. That's because MAX internally checks all IK parameters, finds default values everywhere, then thinks the user never touched them and doesn't use IK. In that case, the MAX IK interfaces are

not even available, so we can't export them ! Let me say it clearly : those default IK settings below won't create a joint because they map MAX's default values.

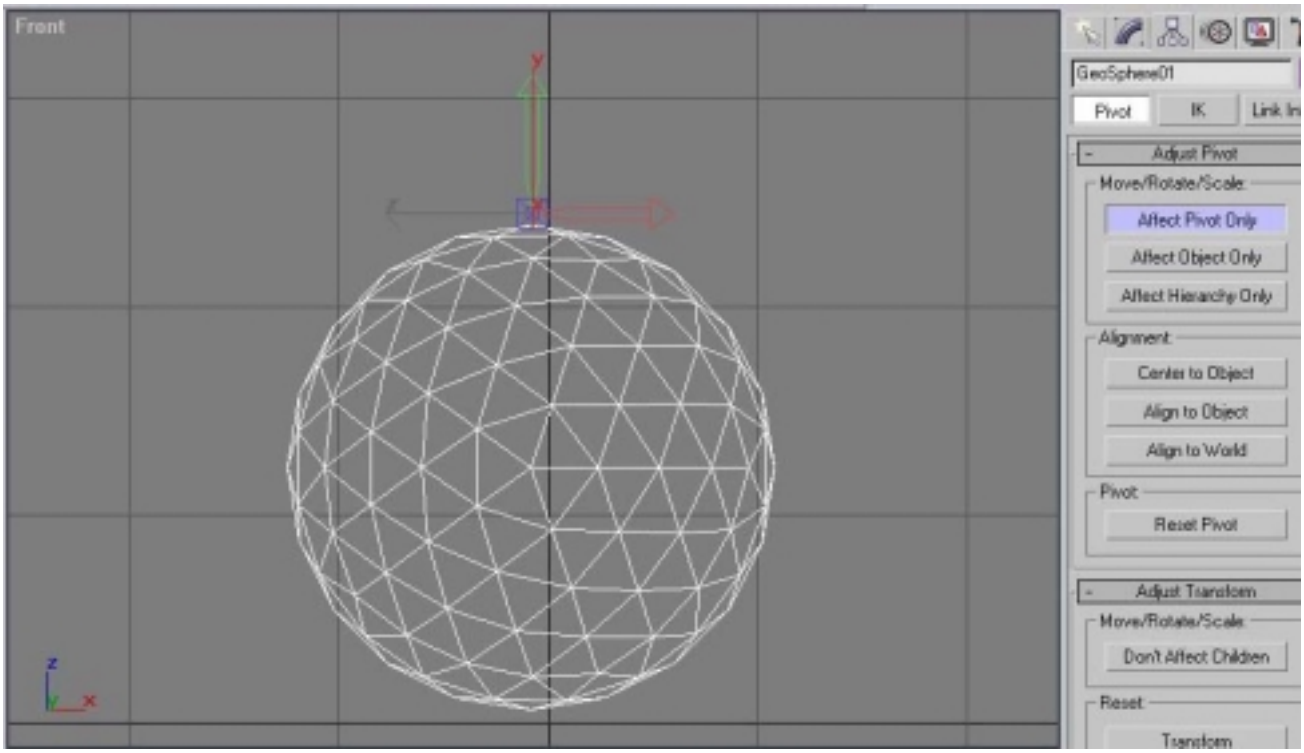


To export the joint anyway, there's a simple solution : tell MAX about it by slightly modifying a value ! For example, just enable **Ease** on an arbitrary axis. Since we don't use that checkbox anyway, it won't make any difference for us.

Let's illustrate this with a new scene : a little sphere chain made of spherical joints.

Step 1 : creating a chain element

We'll just link several spheres together to make a chain. So a chain element is just a sphere. As before, move its pivot so that it's a bit higher than the sphere, out of it.

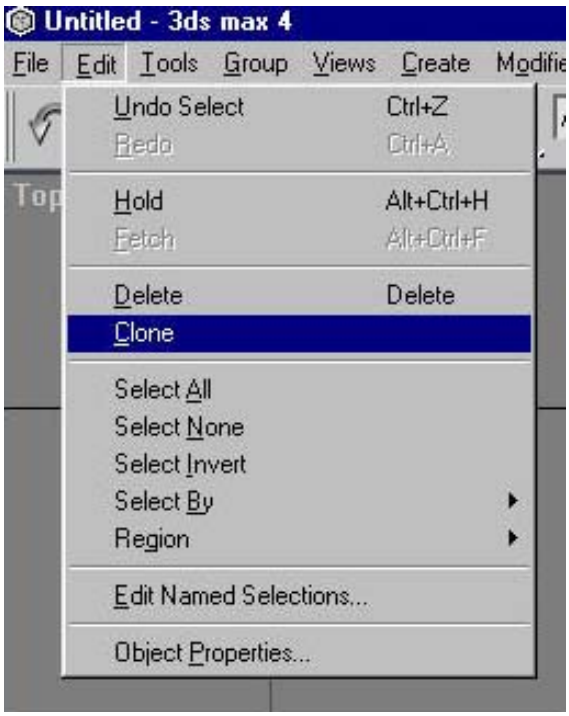


Step 2 : cloning remaining chain elements

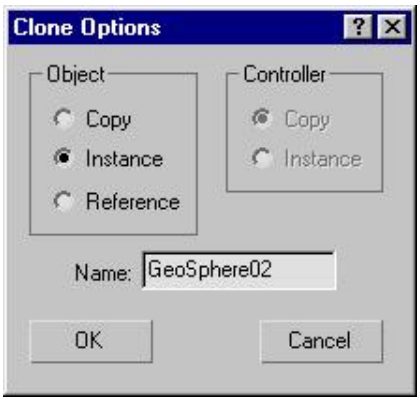
We'll use instances to create remaining chain elements, so that :

- we don't have to set the pivot for all of them
- all pivots are placed in a consistent manner
- the final simulation runs faster

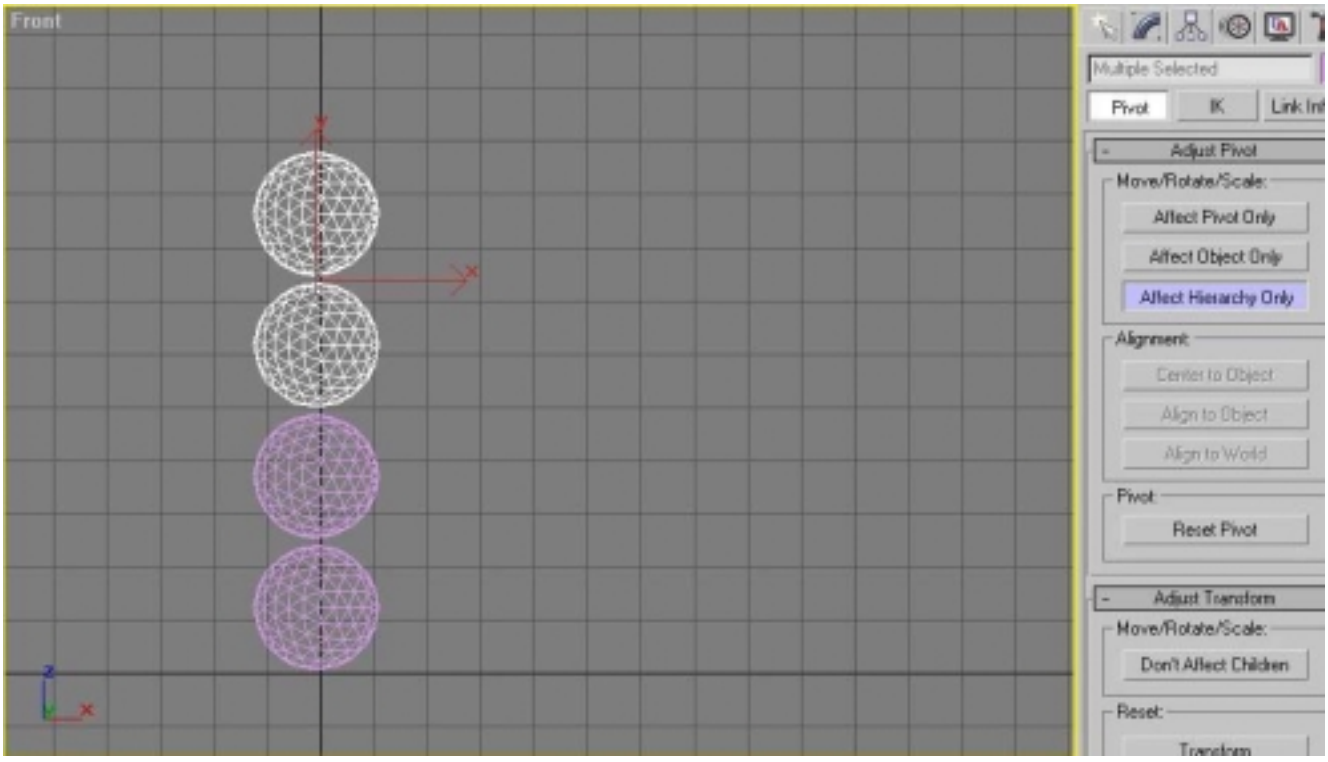
To clone the master sphere (the master is the sphere we just created), select it, then click on **Clone** :



Then create an instance or a reference :

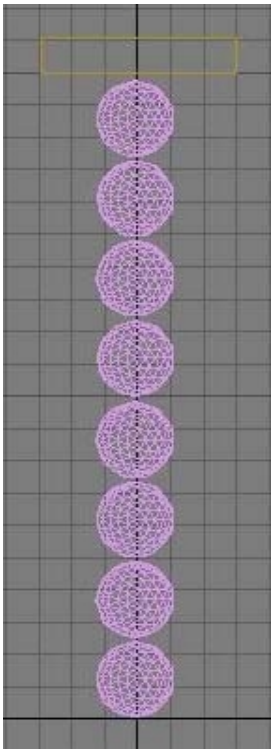


The instance will be created exactly at the same position as its master, so you won't see it at first. Move it and place it above the master so that it starts forming a chain. If you're still in the **Pivot** menu, make sure you're not moving the object *without* its pivot by selecting **Affect Hierarchy Only**. Repeat this several times with new instances. In the image, our chain is made of 4 elements.



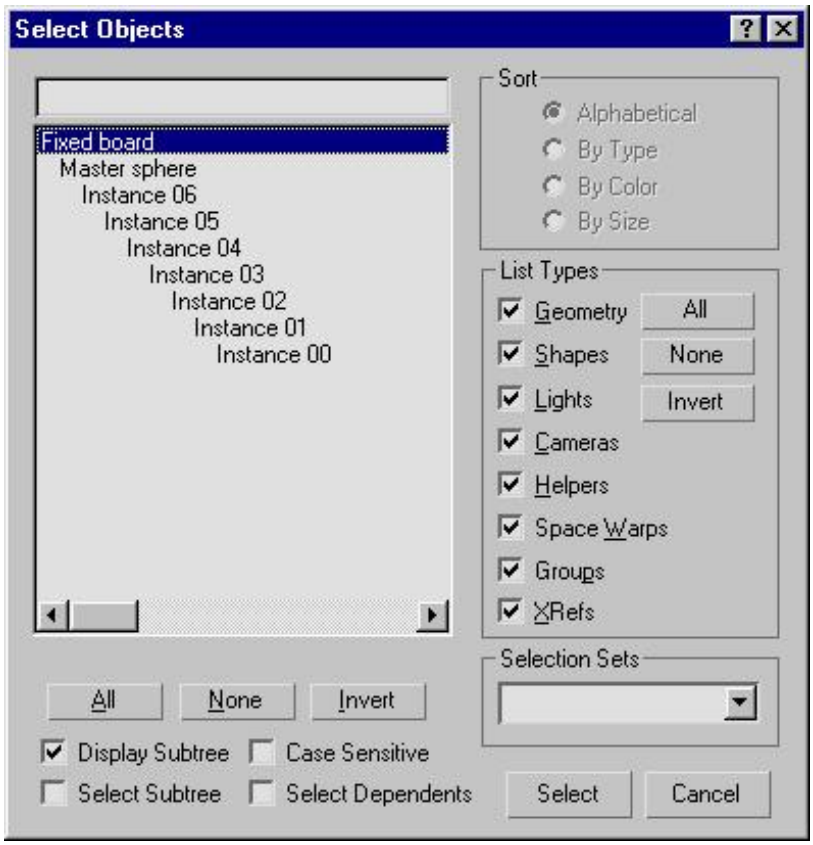
Step 3 : creating a fixed base

Our chain must be fixed to something, else it will just collapse because of gravity. So create a scaled box, like a board, on top of the chain. Set its mass to 0 using the User Properties.



Step 4 : linking all elements

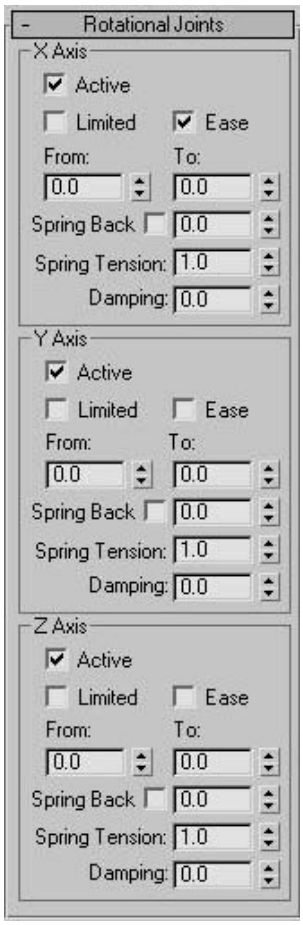
Link all elements in a parent-child relationship. Make the fixed element the root of the hierarchy.



Step 5 : setting joint parameters up

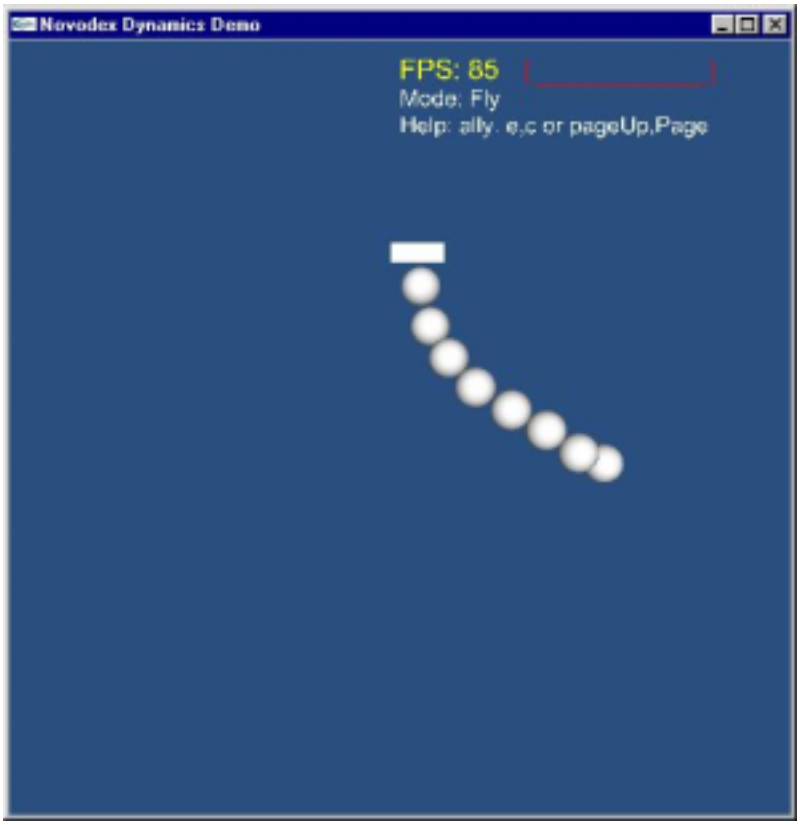
If you enter the IK panel, you'll notice that joint parameters seem to define a spherical joint already. That's what we talked about : those are MAX default parameters. Nonetheless, if you export the scene now, you'll see that the exporter finds no IK data at all ! This is the weird little problem we spoke about.

Solving it is extremely easy : just **Ease** an arbitrary axis on all chain elements, for example the **X Axis**. (touching any other IK parameter works as well)



That's it !

You can now export the scene, and play with the little swinging chan. You should get something like this :



Step 6 : setting bounding volumes up

By default, the exporter uses a bounding box around each body, to perform collision detection. Our chain works, but each sphere is actually bound by a box. You can check this by pressing the B key in the viewer : grey volumes are the shapes used in collision detection. In our case, we should get boxes.

We can obviously do better than that, since we know our bodies are spheres. So we'll tell the exporter about this, by writing the keyword "SPHERE" in the User Properties of the master sphere. All instances will automatically use a bounding sphere as well.

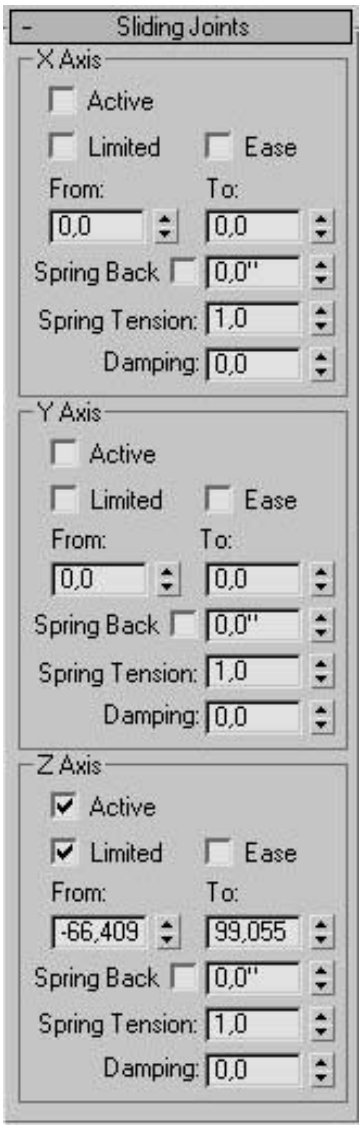
Export the scene again, press B. You should now see bounding spheres around each chain element.

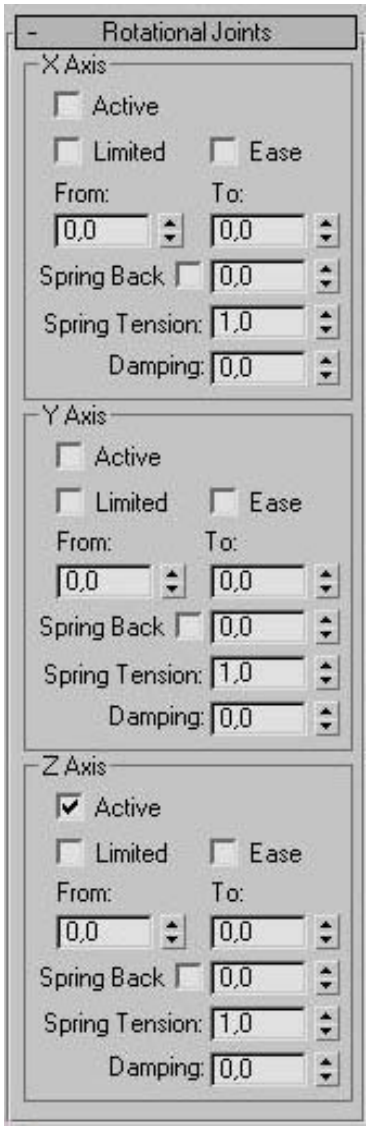
III) Creating a slider

A *slider* (or *sliding* joint) is just a prismatic joint + a hinge. That is, a mesh can slide along an axis, and rotate along that axis as well. The only thing to watch out is that sliding and rotational axes must match. For example, if you slide along the **X Axis**, you must also rotate around the **X Axis** (else it's not what we call a slider).

The joint creation itself should be easy now : if you know how to create a prismatic joint, if you know how to create a hinge, then you know how to create a mix of both. Just change the IK parameters as explained above.

The images below for example, show a slider on the **Z Axis**.



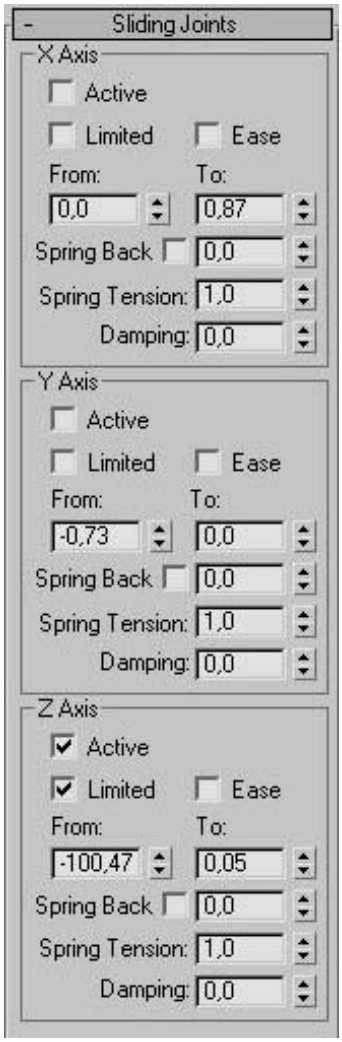


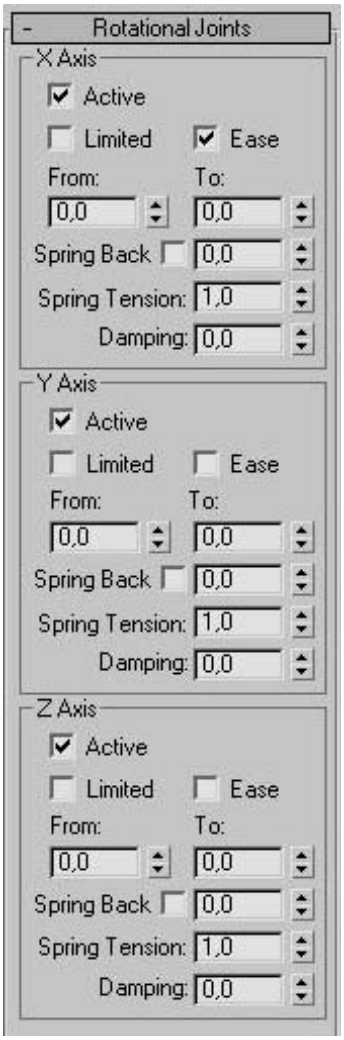
IV) Creating a point-on-line joint

A *point-on-line* joint is just a prismatic joint + a spherical joint. As with the slider, you should be able to create one by just enabling corresponding IK axes.

The "line" part of the joint is then aligned with the **Active** axis of **Sliding Joints**.

The images below for example, show a point-in-line aligned with the **Z Axis**.





V) Creating a point-in-plane joint

Not exported correctly yet !

VI) Unknown joints

Not all combination of active DOFs map a corresponding *NovodeX* joint. Some combinations are not supported, and no joint will be created for them. However, you'll get a warning message in the export window saying an *unknown joint* has been found.

VII) Conclusion

In summary, the only differences between one joint or another come exclusively from:

- the **Active** checkbox
- the **Limited** checkbox
- the **From** slider
- the **To** Slider

...which are parameters of

- the **X Axis**
- the **Y Axis**
- the **Z Axis**

...which are DOFs of

- **Sliding Joints**
- **Rotational Joints**

...which are sub-panels of the IK panel.

Everything else in the IK panel is currently unused by the *NovodeX* plug-in.