FATE - TUTORIAL FOR 3DS MAX v.1.5

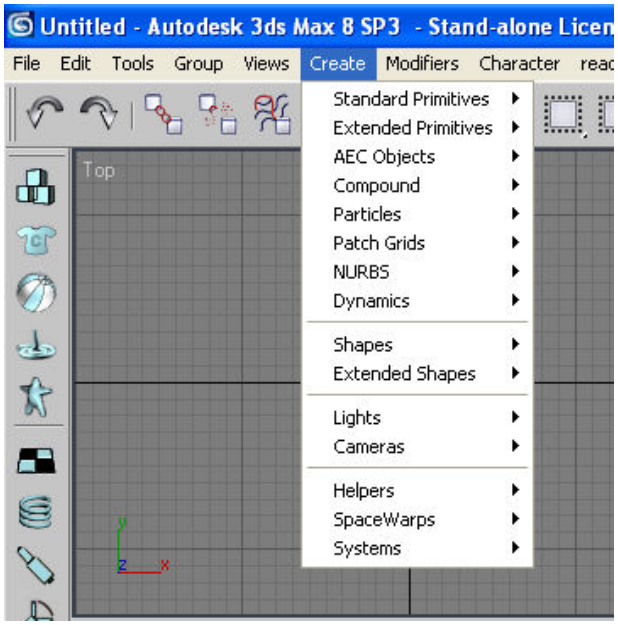
CONTENTS

FATE - TUTORIAL FOR 3DS MAX v.1.5

This tutorial will explain you how to create a simple monster for FATE. You may apply the techniques you learn in this tutorial to more complex scenarios. Make sure you read through the tutorial before attempting to create any complex characters.

1. Creating a mesh:

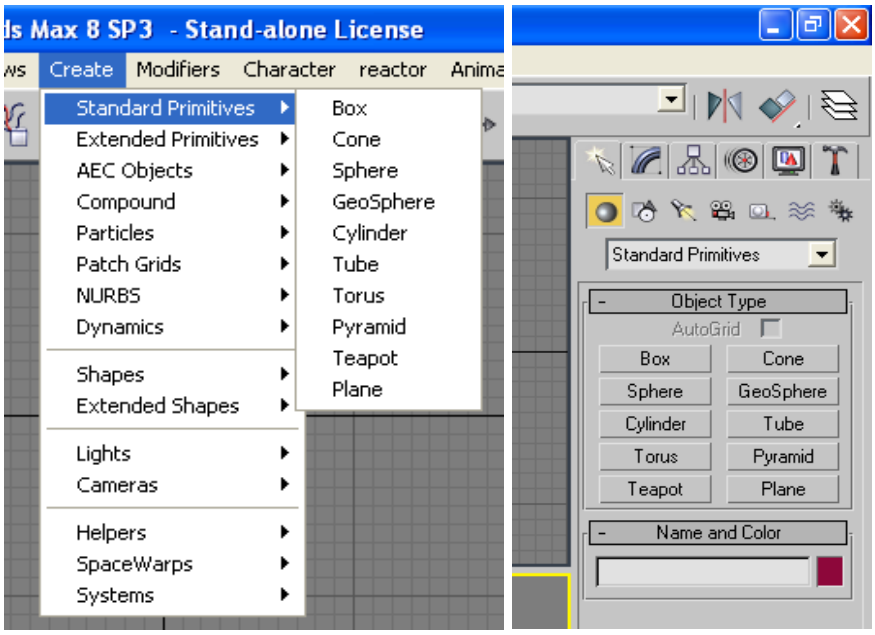
To create a mesh, the 3D representation of your model, you may select several of the options from the **Create** Menu.
The two submenus you will use most often will probably be the **Standard Primitives** and the **Shapes**.



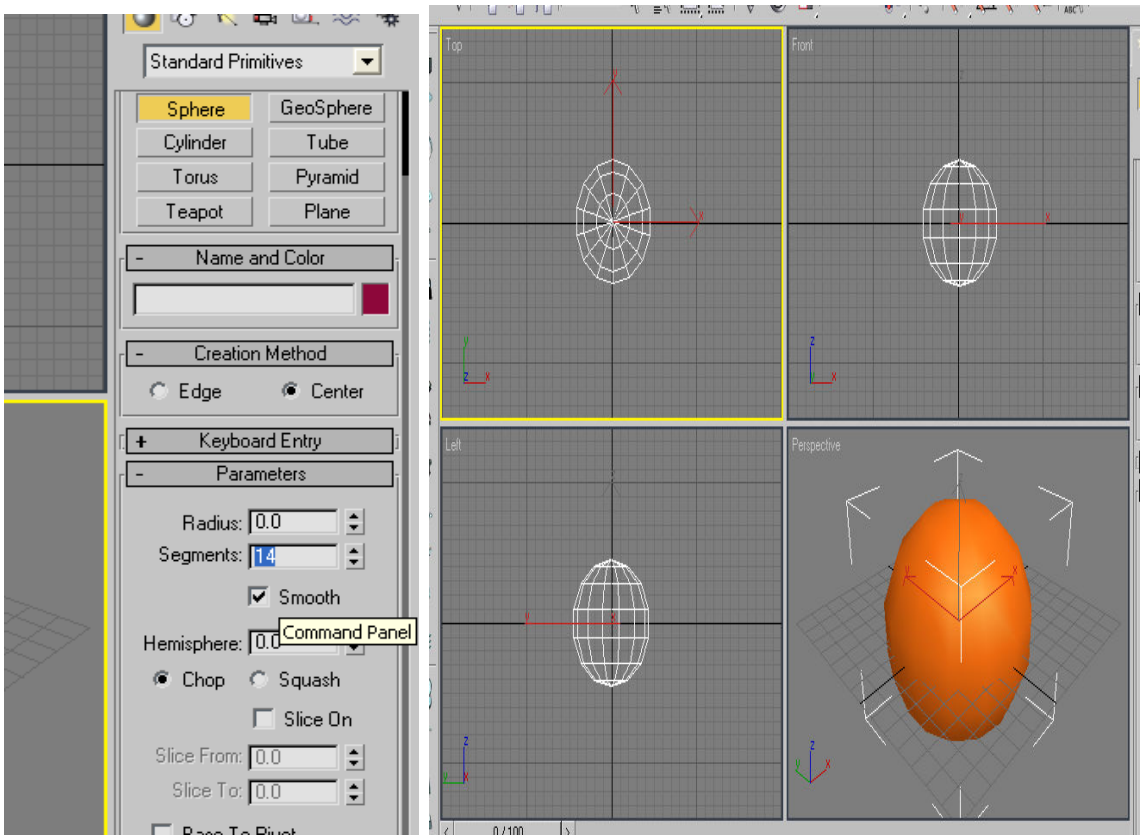
a. Standard Primitives

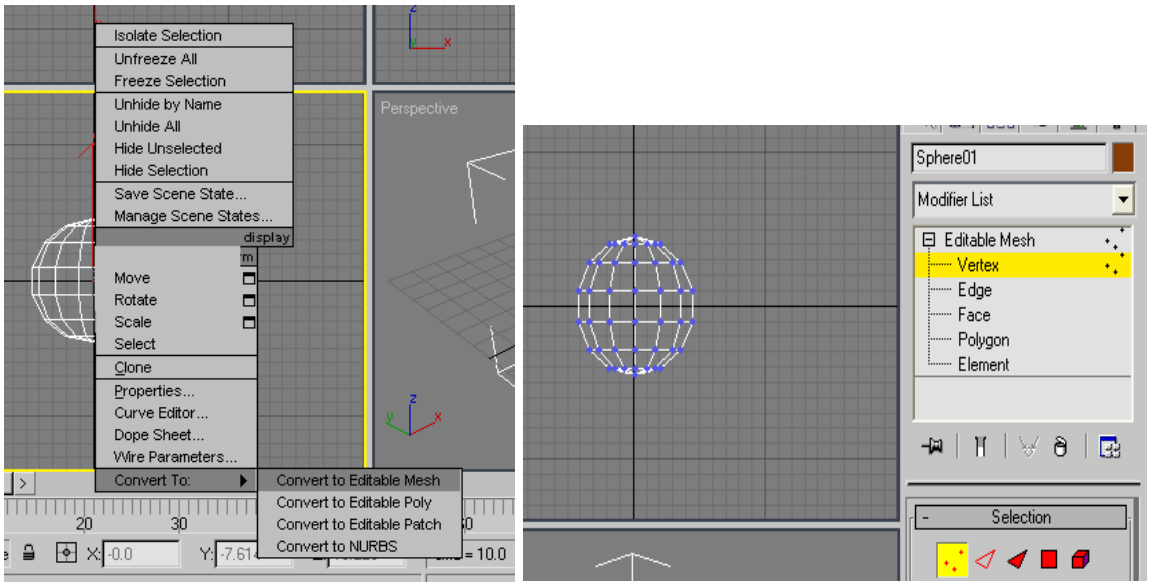
You may choose to create your character by using a standard primitive. Like clay you may shape your sphere to look like a face, or a whole character.

For this tutorial we will use a **sphere**, and convert it into the main body of a monster.

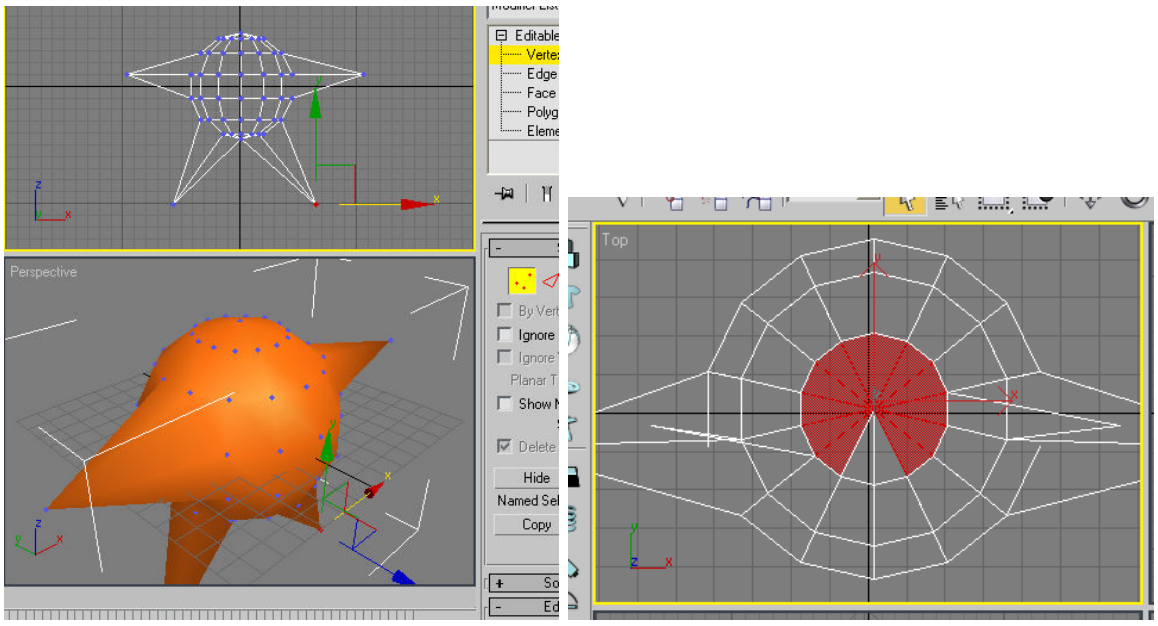


From the primitives panel I have selected **Sphere** and from the side command panel I have changed the number of segments it will be made up of to 14 (keeping the number of faces and verteces to a minimum is desired to minimize in-game processing time once the model is finished). Clicking on the top viewport I have created my object.

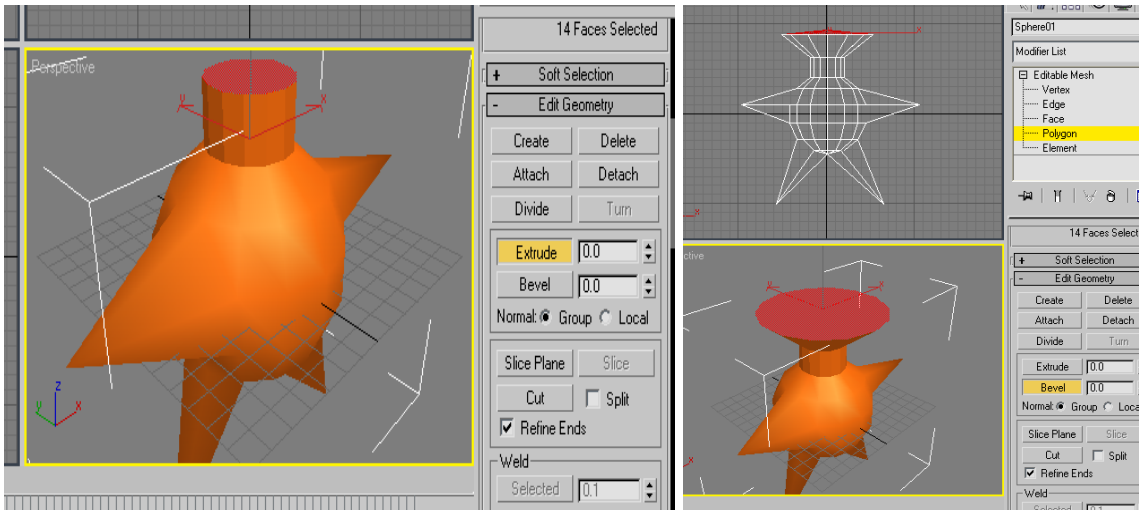




Right Clicking on my mesh I select to convert to **Editable Mesh**, so that I can start manipulating it. When I select vertex, I can see my sphere's points which I can work on.
By right clicking on the mesh I can **Rotate** and **Move** the verteces as I please. There goes a simple mesh.

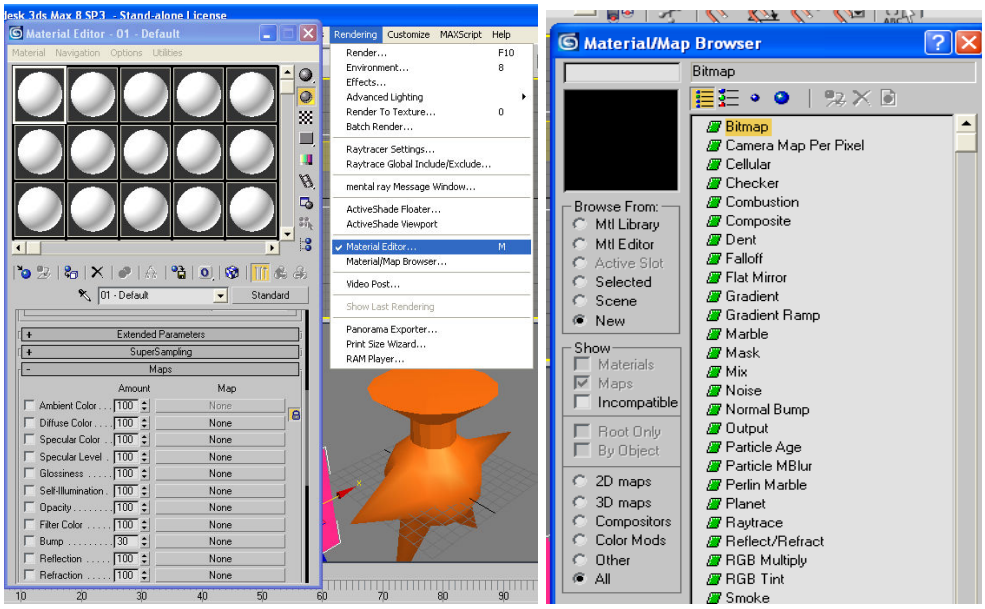


I will select the mesh by polygons this time and will add new faces to my object to form a face. In the command panel on the right I have selected the **Extrude** function so that I would be able to add a new component to my mesh. I can do something similar with the **Bevel** tool to create a sort of hat for my character.

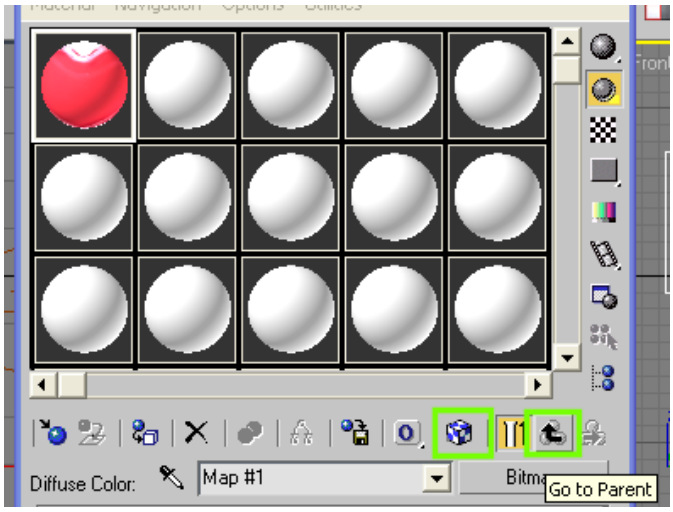


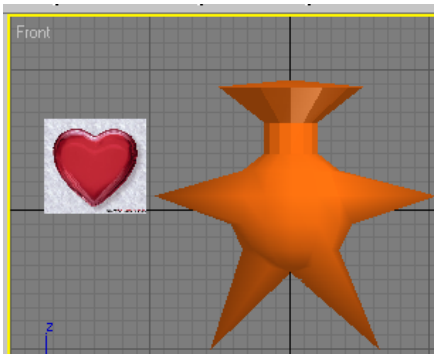
b. Shapes

With the first part of my model ready I can now move on to creating another component of the character by using the shapes. I am not very good at freehand with the mouse, so I will create a heart by helping myself with a ready made pic.
I will first need to create **panel** mesh on to which to add the pic I'm going to work on. I will do this through the Standard Primitives menu. Once done I will upload my picture onto max. Pressing **M** will call the **Material Editor** window, once inside, scrolling down and opening the Maps tab will reveal a command panel as the one below. Clicking on the empty slot near **Diffuse Colour** I will display a **Material Browser** so that I can select my **Bitmap**.



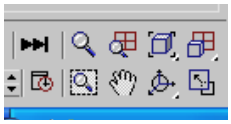
Once selected the picture, I will **Go To Parent** panel and tell max to **Show Map In Viewport**. I can then drag my map onto the panel so that it will be associated with the mesh. This is a very simple way of loading your map onto the mesh. Note that in the picture below I have changed my front view to display as **Smooth + Highlights**.





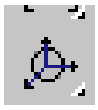
Now that I have a picture to work with I will create a **spline** (a line mesh) through the Create > Shapes > Line function. In the side panel under **Creation method** I will select a **Corner Drag Type**, so that I will have rigid segments without curves.

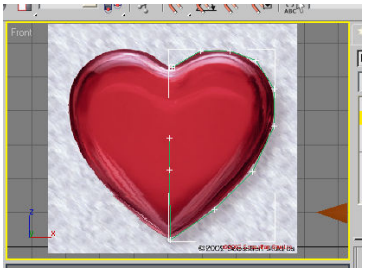
Note:



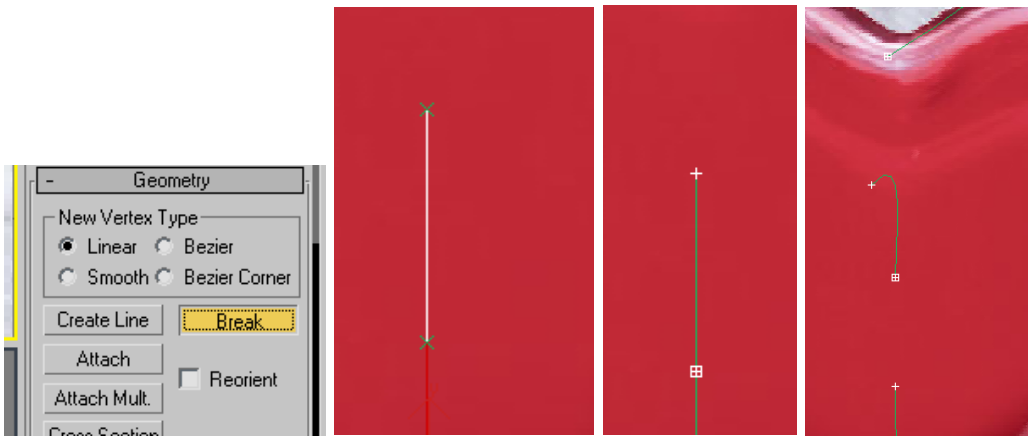
the functions at the bottom right-hand corner of the screen allow you to modify the view.


the Hand tool lets you move your view focus

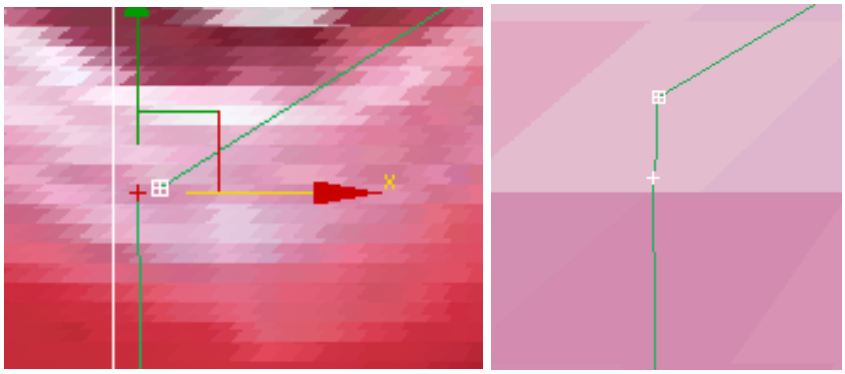

the Rotation tool lets you rotate your field of view.



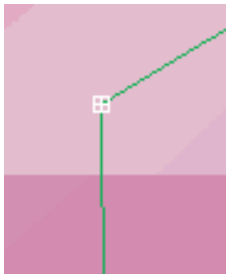
I have only created half a heart as an example. I can modify my splines vertices much like I did with the mesh. I will remove the extra line in the middle by the **Break** function in the side panel and then delete it.



I will then connect the two points to make a continuous spline. I need to drag the two end verteces towards each other, select both of them when they're overlapping and choose **Connect** from the side panel. I might need to zoom in a little.



Connecting the two ends together has produced an extra segment I don't need. Any two verteces I select of a same object can be fused into one through the **Weld** tool. I will set the threshold level for the distance separating any two vertices, select the two vertices and click weld to join the two points into one.

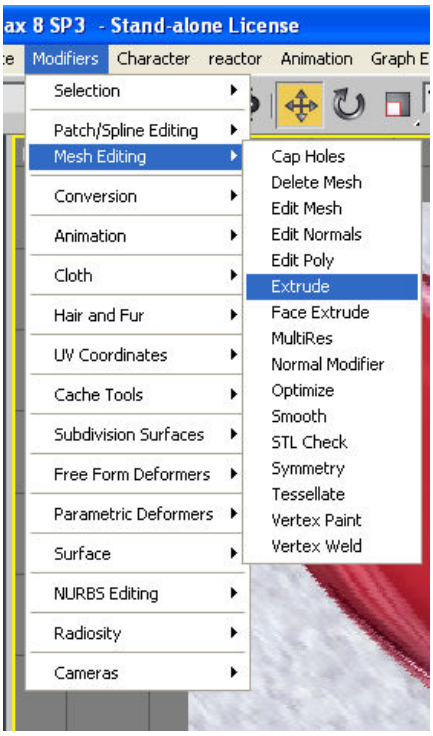


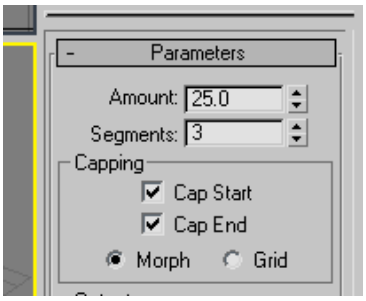
NOTE: There are several other functions in the **Edit Geometry** rollout; the most useful will be displayed if you convert. Your mesh to an **Editable Poly**. You will find yourself needing to manipulate different components of your mesh such as verteces, edges, faces and polygons. Functions like Bridge Edge or Polygon will let you close gaps; Delete Edge will let you fuse two polygons into one leaving all their map coordinates intact; Insert polygon will let you create a smaller polygon within the one selected, while Tasselate polygon will break your polygon into smaller parts.

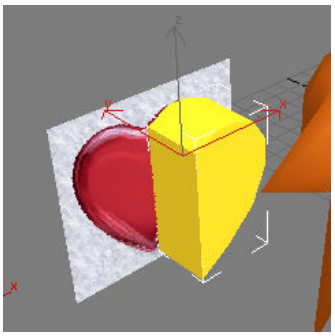
2. Modifiers

a. Extrude

The first half of the heart is ready. To convert the spline from a linear shape into a 3D model I will make use of a Modifier called **Extrude**. I can set the number of segments I want to subdivide my 3D shape into and the width amount.

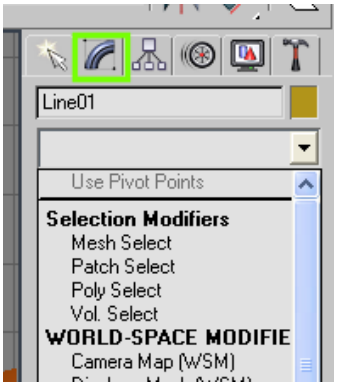




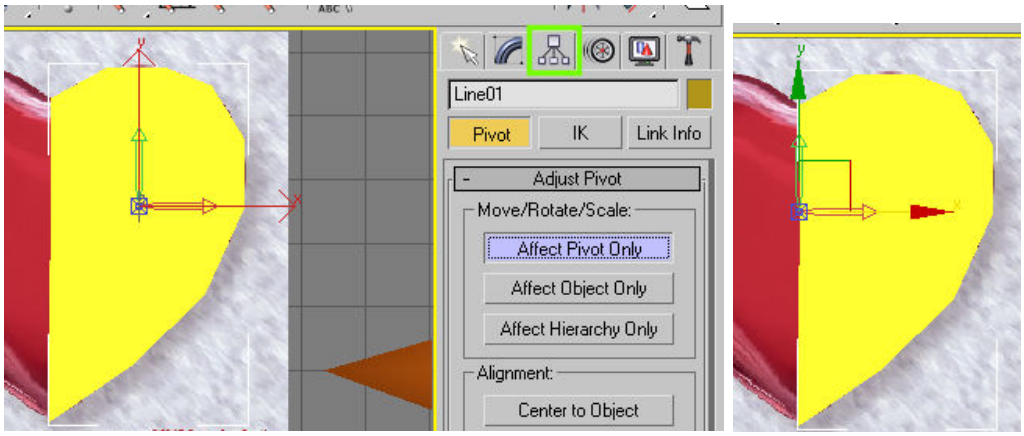


b. Symmetry

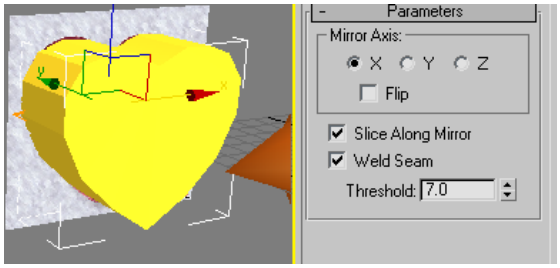
I can create the second half of the heart by using another modifier, named **Symmetry**.
This is easily accessed through the side panel's drop-down list.

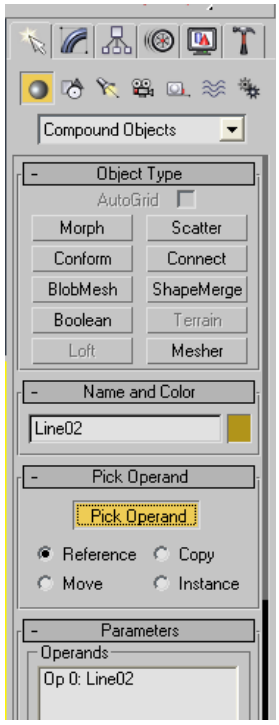


Before I can use the symmetry modifier effectively, I need to specify where the mirror plane is. I can do this by repositioning the object's pivot at the edge of my half-heart.



In the hierarchy panel I will select **Affect Pivot Only** so that I can move my pivot as any other object.
I can now perform symmetry, choose the mirror axis and to weld the two halves at the interface.

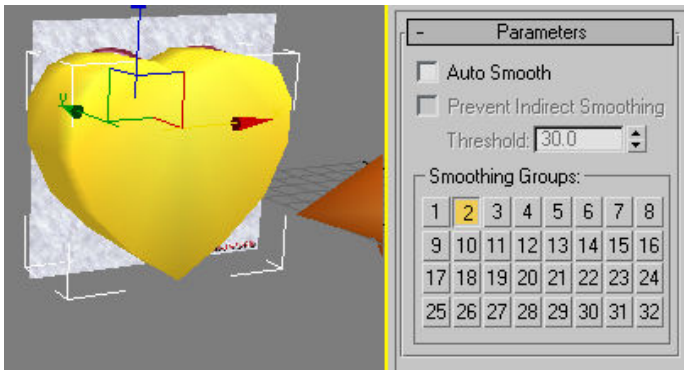




Note:
I could have obtained a similar result by creating a **Mirror Clone** through the **Tools** menu, but this at times produces incorrect results.
I would then have had to connect the two components through Compound Objects > Connect > Pick Operand and clicking on my other component. I could then have welded the verteces of the two parts. This manual method sometimes produces errors but it's valid still.
Separate polygonal meshes can also be joined through the **Attach** function in the Polygons submenu.

c. Smooth

I now have a complete heart but looks too square, so I will smoothen its edges through the **Smooth** modifier. This is simple to use and I only need to select any smoothing group to obtain the desired result. You can select different polygon sets and assign different smoothing groups to them so as to make them smooth yet maintain the separation of the two parts; this is useful for modeling the lips of a mouth for instance.

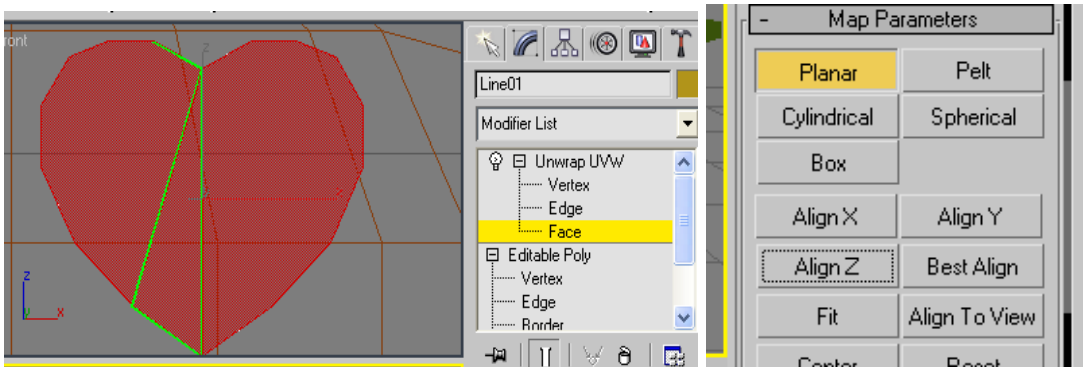


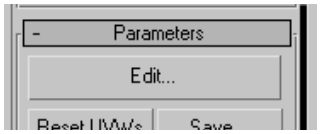
3. Mapping

a. Planar Maps

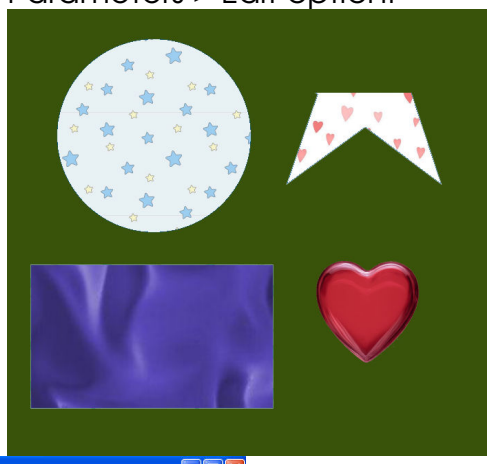
All the objects' meshes are finished and can be assigned material maps. I have prepared a map I will use for all of my meshes, I can select different parts of my picture to be assigned to different components. To do this I will first associate the picture to the two different meshes as I did before with the panel object.

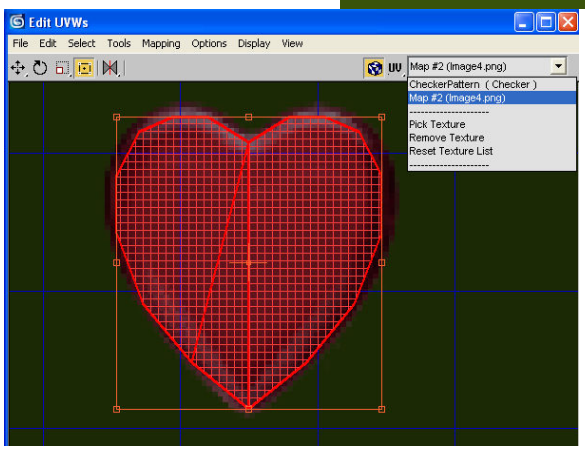
I will first map the heart. In the modifier list I select **Unwrap UVW**, and in the object's mesh I select the top faces of the heart. In the Map Parameters panel I select Planar > Align to Z, to align my map's plane with the top face of the object. Make sure you deselect the Planar command afterwards.

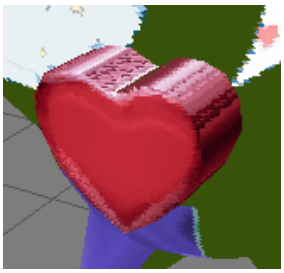




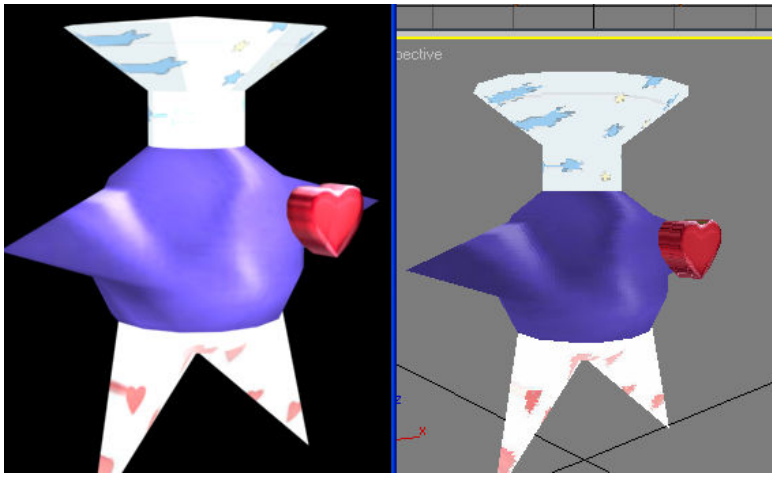
I can now edit how the map is set through the Parameters > Edit option.







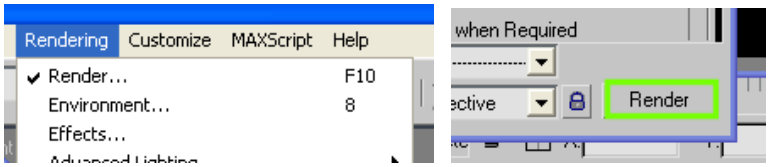
Once in the Edit UVW window I have loaded my map through the scroll down list on the right. I can now position my mesh to overlap the picture of the heart in my texture map. As you can see the side faces of the heart have a distorted map.
Sometimes you may need to move the verteces of the UVW map so that the separation between the frontal and side faces is increased.



The procedure for the other mesh object is the same.

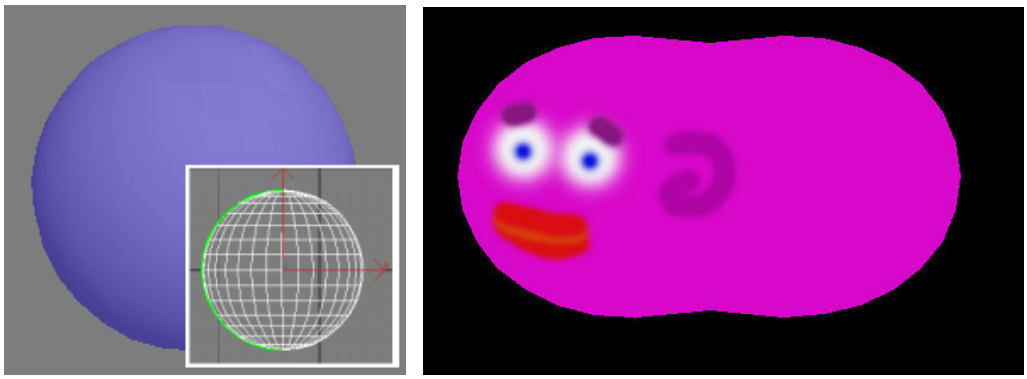
The resulting map in the 3ds max viewports may appear degraded but this is

simply because you are seeing a rough preview, so don't be deceived. You can see the actual result through the Render menu, and select the Render option.



b. Pelt Maps

For more complex mapping, such as for the face, you may want to use a Pelt map instead of a Planar one. This is slightly more complicated. We still apply an **Unwrap UVW** modifier.

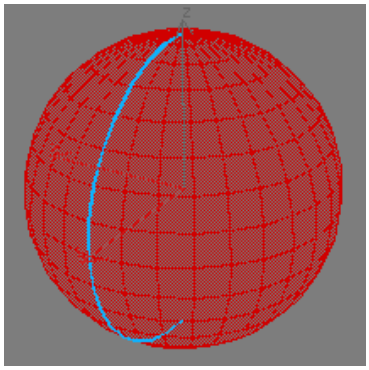


Consider the following simple object and its map. The green line at the side of the sphere is a seam line which max creates by default. This specified where the object would split if we were to unfold it. We are however going to work with custom-made seam-lines (eventhough in a simple case as this case the final result will be the same as that of the default).

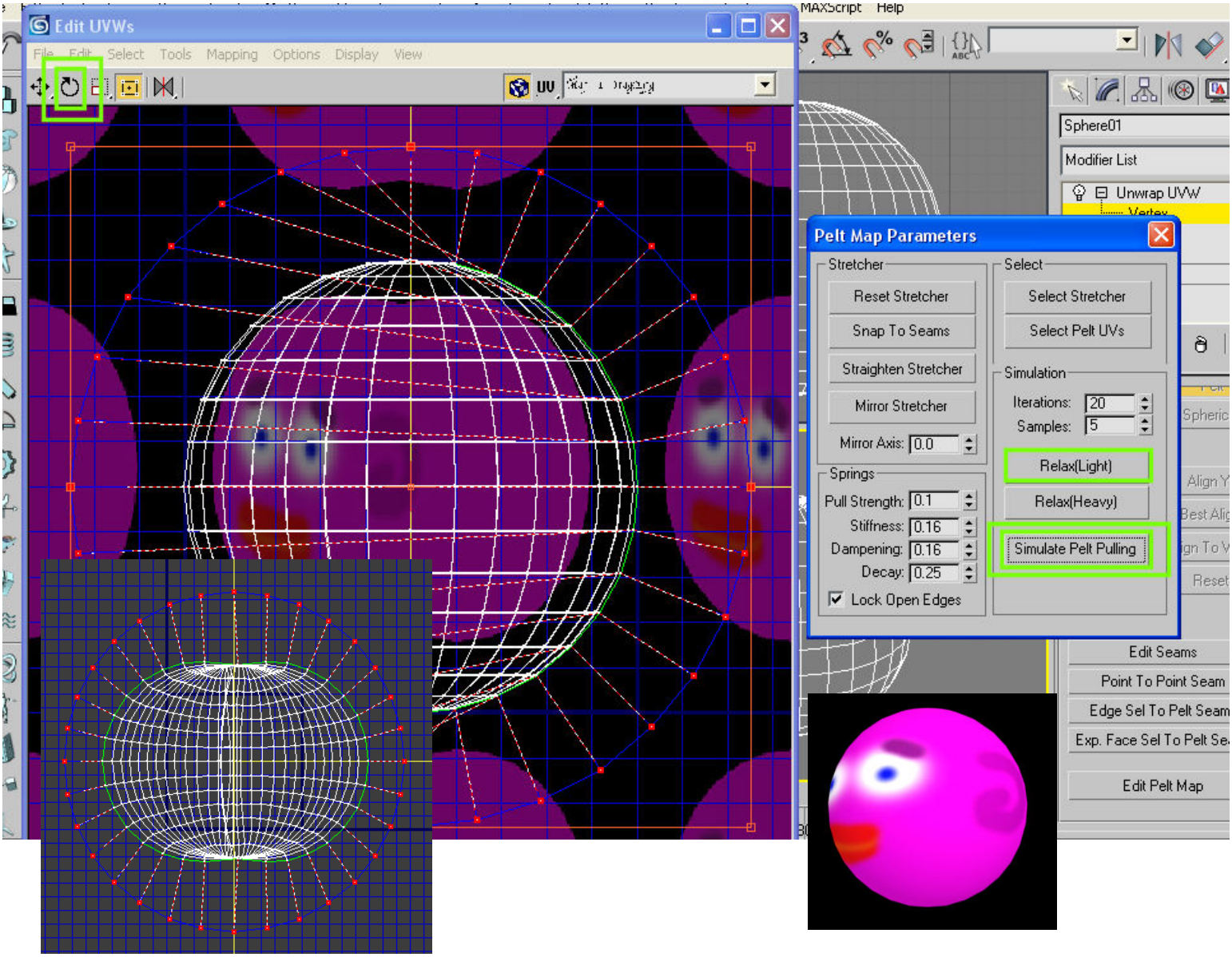
To avoid confusion, the default seams can be disabled through the display menu in the Unwrap UVW modifier, by selecting **Show No Seams**. Also check the **Always Show Pelt Seam** checkbox.

Selecting the Face submenu and scrolling to the **Map Parameters** rollout, select the **Point to Point Seam** option to be able to trace your own splitting lines on the object's wireframe.

The Pelt Seams will be blue, as shown below. Once all your polygons are selected and the Pelt Seam line is marked you can select the **Pelt** option, align the **cross-section** accordingly, and click on **Edit Pelt Map**.

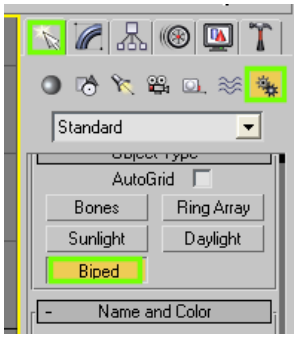


You are taken to a Pelt Map Parameter pop-up, and the Edit UVW window. First **Rotate** the pelt pulling strings so that most of them are symmetrically aligned around the object's map. Then click once or twice on the **Simulate Pelt Pulling** and a few times on **Relax (Light)** to stretch your map. The you simply need to overlap your pelt map on your UVW map.

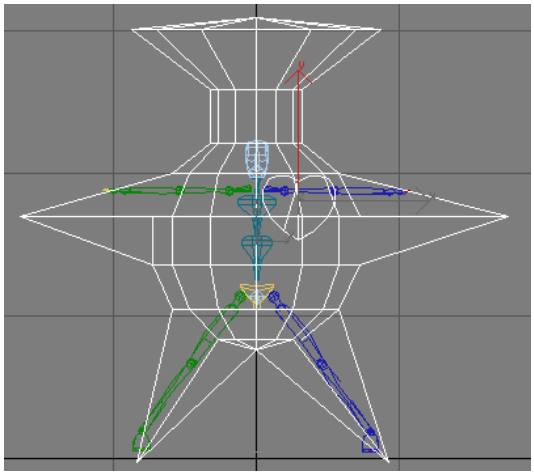


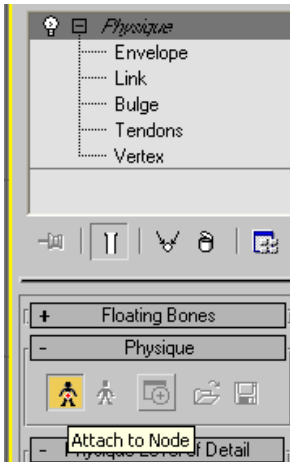
4. Adding Bones

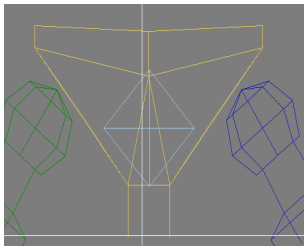
a. Using Biped Bones and the Physique modifier



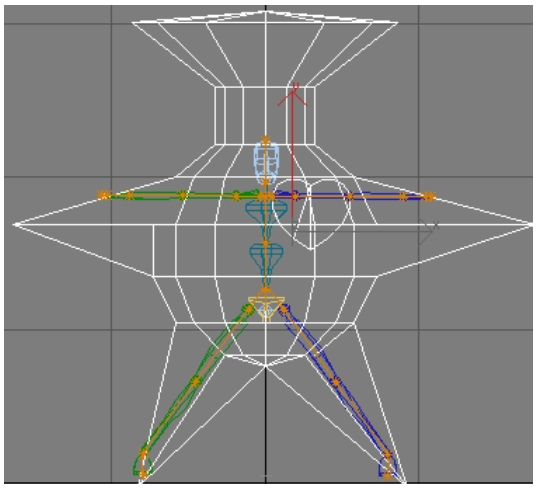
Through the **Systems** option in the side panel I can make use of **Bipeds** and specify what they will be like. I will select a simple skeleton without toes nor fingers, and only two spine segments enough for it to bend. Create the biped in the front viewport and try to make it take the shape of the model as much as possible. Make sure your skeleton and your character are facing the back viewport since the perspectie in FATE is inverted.



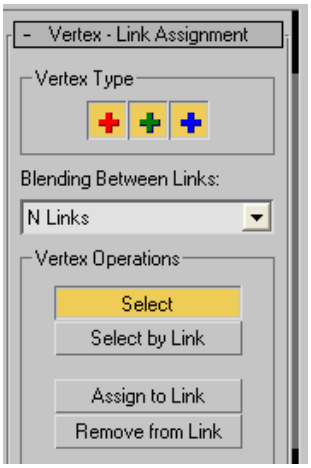




I will select both the meshes and choose the **Physique** modifier in the side panel. After selecting **Attach to Node**, by clicking on the **Centre of Mass** (the diamond-shaped bone in the pelvis) and clicking on **Initialize**, I will associate the skeleton to the mesh. The result will appear as an orange stick figure inside the skeleton.



Supposedly once the skeleton is linked to the mesh, the verteces should be assigned automatically but this is not always the case, so it is convenient to assign the verteces manually. To associate the verteces to the skeleton of one of the meshes, I will select the mesh I'm interested in select Vertex under the Physique entry and will be able to access a panel as the one below.



The functions are self explanatory.

Red cross: Deformable verteces
Green cross: Rigid verteces
Blue cross: unassigned verteces

By excluding either you can see which verteces have been assigned and which not and to which link. You can also unassign the selected verteces to links, by highlighting only the blue cross and selecting **Remove from Link** from the commands, and then clicking on any bone link (orange lines).

Link verteces to the desired links by selecting the desired verteces, clicking on **Assign to Link** and highlighting the Green cross. Click on one of the links to complete the task. Once assigned, the vertices should turn from blue to green. You may assign a single vertex to more than one link. In that case the vertex points will appear of a darker hue.

When the links have been assigned you may click on the skeleton to move your model, at which point you may proceed to creating animations.

Note:
If I desired to add additional bones I may have done so through the **Bones Tool** in the **Character** Menu. The width, height and taper may all be specified. To associate non-biped bones (IK bones) to the mesh you will have to link them to your biped and then reinitiate your physique modifier. The IK bones may be linked to the biped skeleton through the Link function on the main command panel.

This will make sure that as your biped skeleton moves, your other bones will move with it. This may also be applied to different meshes.

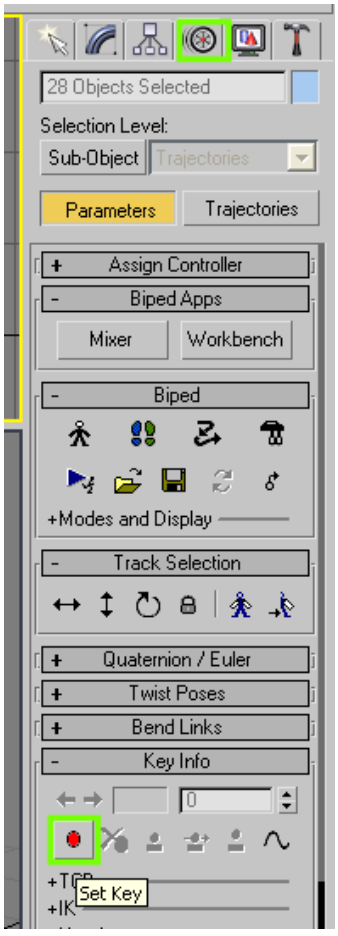
b. Using IK bones and the Skin Modifier.

You may also create and animate models by using exclusively IK bones. These have no constraints like the biped bones and are very versatile. You can even animate your model to shrink or bend in unnatural ways, also bones may be placed at any coordinate and need not be adjacent to any other bone of your skeleton.

I use the **Skin** modifier to link the mesh to these types of bones. This works in a simpler way than the biped.

The mesh may be associated by selecting the desired bones through the **Add** option and the bones influence **Envelopes** may be rescaled or repositioned. Otherwise, the influence a bone has on each vertex may be specified manually through **Weight Properties** or the **Weight Table**.

Remember that the influence fractions always add up to one. Assigning envelope values manually produces less realistic results than doing the same through envelopes unless you know what you're doing! Nonetheless manual assignment may be useful for rigid characters as robots.

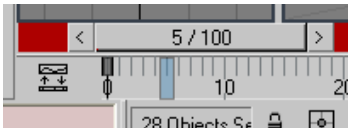


The Skin modifier gives the advantage that the verteces are connected to bones through gradients of influence and thus produce better flexing joints. This advantage may also be made use of with biped skeletons instead of the Physique modifier, however if you're going to use the Skin modifier make sure that your mesh is of the correct size as you want it to appear in the game, otherwise you will need to restart the bone-connecting part all over again just to resize your model.

5. Animating a model

First thing to do when animating is to freeze your main meshes to prevent them to get in the way. You can do this by right clicking on the selected mesh and selecting **Freeze**.

To animate you need to turn on **Autokey** in the bottom panel. You will use **Rotation**, not Move. For the biped skeleton you need to set starting positions however this is not necessary with other bone types.
To set starting positions you either rotate biped and replace it in position or you may select all the bones of the biped skeleton and in the Motion panel (the wheel icon) select **Set Key** in the **Key Info** drop down panel.
To create looping animations you may copy the starting position to the last frame by clicking on the key in the timeline at the bottom and drag it to the last frame while holding down **Shift**.

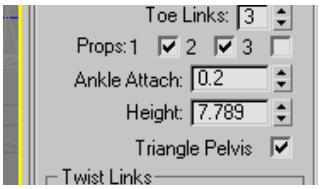


From then on, whatever bone you will move on the other frames, the skeleton will automatically be made to move from its starting position to the final you have assigned to it.

Trajectories:
To move the biped skeleton in space you need to click on the **Trajectories** option in the motion panel. Move only the centre of mass. You still need to set a starting position for the trajectory, but the rest is as simple as it is.

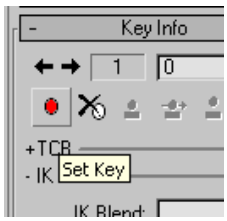
6. Adding Weapons

a. Prop Bones



The location of weapons on FATE's **Main Character** is specified through special bones called **Props**. You can add these though the Biped's Structure rollout menu in the side panel. You will see three checkboxes for three independent Prop bones. FATE's characters are made to hold a maximum of two weapons, one per hand, so Prop bone 3 is excluded for this purpose.

When you are in the Structure menu of your biped, you can modify the size, rotation and location of your prop bone like any other bone of your skeleton. Once you exit the structure menu this position will however be reset and you will have to reassign your prop its coordinates.



To make your prop follow your hand bone in a way which is natural to any hand-held object,

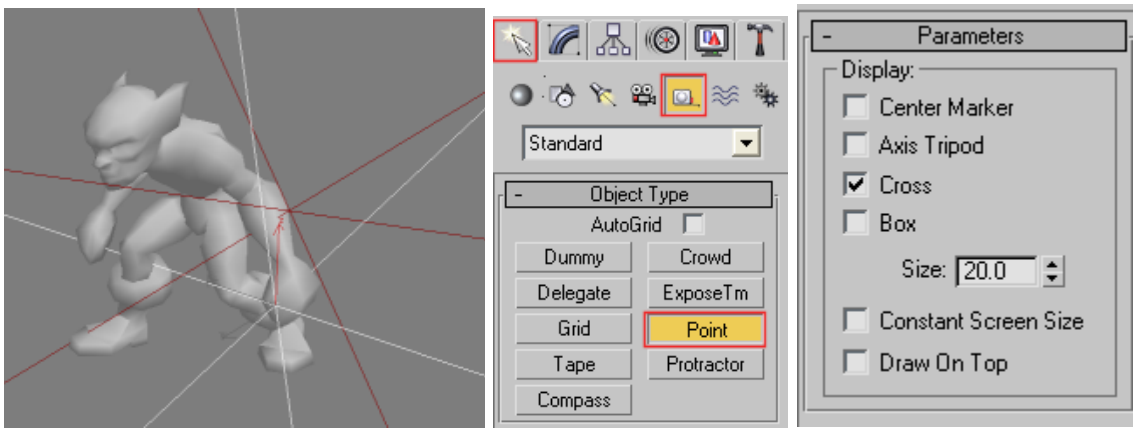
select the bone, and in the **motion panel**'s **key info** dropdown menu **set a key** at frame 0. In the Prop Submenu you will now be able to choose on which bone to lock the prop bone.

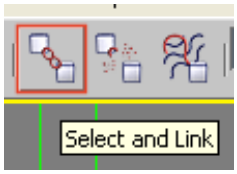
Once you set the **Rotation Space** to **Right (or Left) Hand** your prop will follow your biped's right hand bone naturally. You will also need to add **point helpers**.

b. Point Helpers

The weapon's coordinates for the **Main Character** and **Monsters**, these are specified as **Point Helpers** and can be created through the side panel under Create > Helpers > Point. Make sure the helper is set to the **Cross** type. Through such an object you can specify the location of left- and right- hand weapons and the shield, set on the left arm.

So that the helpers are recognised by FATE you simply need to name these objects as either "**lefthand**", "**righthand**" or "**leftarm**" respectively and position them accordingly. The main character has an additional helper for the helmets which is to be named "**head**".





Once you have positioned your helper point in place, link it to the respective bone of your biped skeleton through the **link** button in the top panel.
Setting correct coordinates for the helper's orientation is not as difficult as it seems, although this would be a very tedious job if you were otherwise unaware that the orientation of the corresponding weapon is very related to the shape of the helper itself. Consider the helper as being like a sword: the vertical axis representing the blade and the pommel; the intersection point representing the guard; one of the horizontal axes corresponding to the cutting edge of the blade. To

determine which is which you will need to do some experimenting by loading your character in the game.

IMPORTANT:

It is good practice to create all your animations in a single .3ds file instead of separating them, then setting which frames need to be exported for which animation. This prevents that a mistake or an omission made at a very early stage can be easily corrected for all animations at once, especially when dealing with point helpers for the setting of particles or weapons.

When using point helpers remember to leave the **Export Helpers as Tags** option checked if you want your helpers to show.

FURTHER NOTES 1 – EXPORTATION AND IN-GAME RENDERING

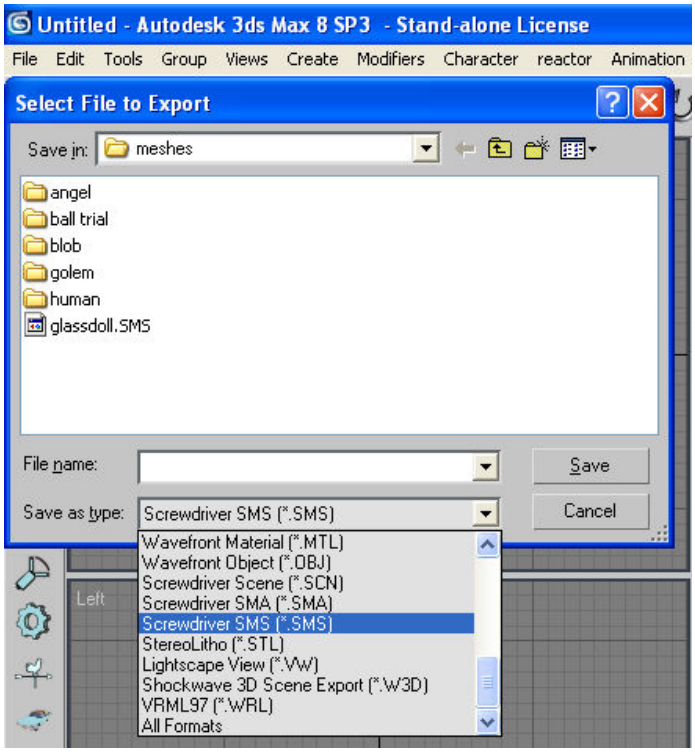
Basic tools:

- 3ds max must be equipped with the MODKIT plugins provided by Travis Baldree (c) WildGames They will let you save your main model's mesh in .SMS and your aniamtions in .SMA format.

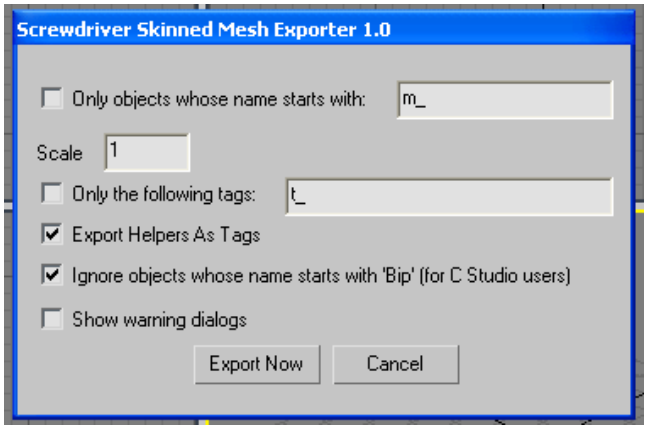
Exportation:
Animated models need be exported as SMS files, and their respective animations need to be exported as SMA files.

You may also export your model as an AMS (animated mesh) file; in this case all the animations need be in a single file. This saves you time from having to export single animations, however without .SMA files you will not be able to transfer skeleton animations to other characters you create which share the same skeleton.

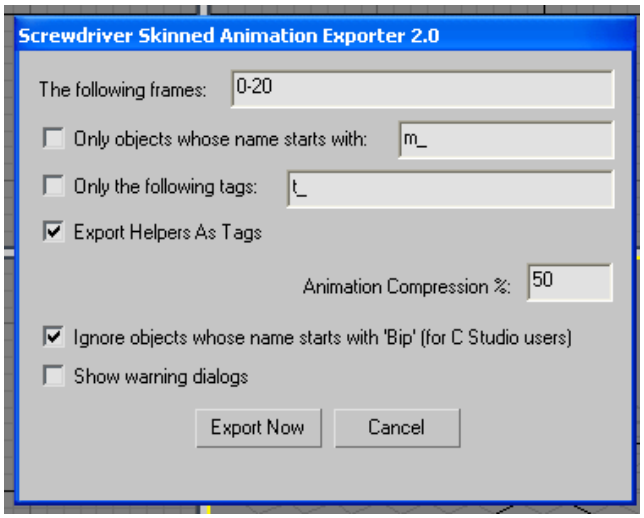
When exporting your finished character, with verteces properly associated to a biped skeleton and all, you will need to access File > Export and save your character's mesh in SMS format.



You will be prompted with a window as the one which follows. You need to disable the "Only objects whose name starts with" checkbox.



You can reduce the scale of your model at this stage; a scale of 0.04 sometimes is reasonable if your model is as large as the entire grid.
Your animations must be exported in SMA format and you will be prompted with the following window. Again disable the "Only objects whose name starts with" checkbox. Specify which frames you want to export; in the example below, frames from 0 to 20 are being exported.



Once you have at least one SMS file and one SMA file, the animations for the SMS file must be referenced in an Animation.dat file, an example of which is provided below, taken from the CAT character in the game.

CAT

animation name : first frame : last frame : FPS
IDLE:idle.sma:30
WALK:walk.sma:30
RUN:run.sma:30

ATTACK1:attack1.sma:30
ATTACK2:attack2.sma:30
DIE:die.sma:30
LICK:lick.sma:30
HUNT:hunt.sma:30
#animation keys <KEY>: animation the key is for : frame of the animation (can be float) : name of key
<KEY>:ATTACK1:18:HIT
<KEY>:ATTACK2:32:HIT

The last two lines (where <KEY>:**ATTACK2**:**32**:**HIT**) specify at which **frame** will the given **animation** result in a **hit**. In this case, attack2, will hit the target as the animation attack2.sma will reach frame/key 32.

To load the model in the Viewer application supplied with the MODKIT or the game itself, make sure that all SMA SMS COL MDL files are available and present in the same directory, otherwise the model will fail to load.

For each model, so as to prevent the game from crashing and to ensure the game loads as it should, the essential components must be:

+ Animation.DAT (containing all the information concerning the SMS file and its related SMA animations)

+ collision.MDL (an object defining the model's boundaries - can be a simple cylindrical mesh around the character)

+ youranimation.SMA (animation files)

+ monster.SMS (model file)

+ BMP or JPG texture map files

For the SMS model to load without any missing faces, all of the mesh's vertices must be attached to at least one bone in 3DSMAx. (In case the term is new to you, the mesh is the 3d wireframe).

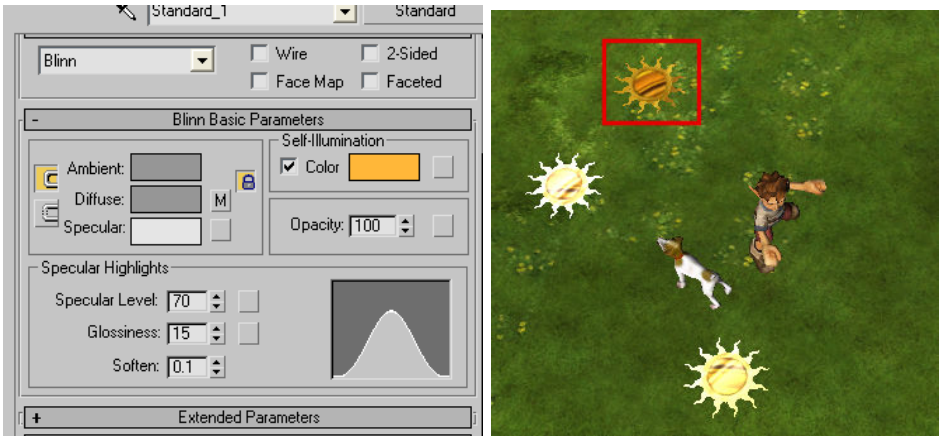
The collision file is essential as any other component, so make sure you include it, although the game may still load and the character may still be collided with, the game may crash after a while if this file is missing.

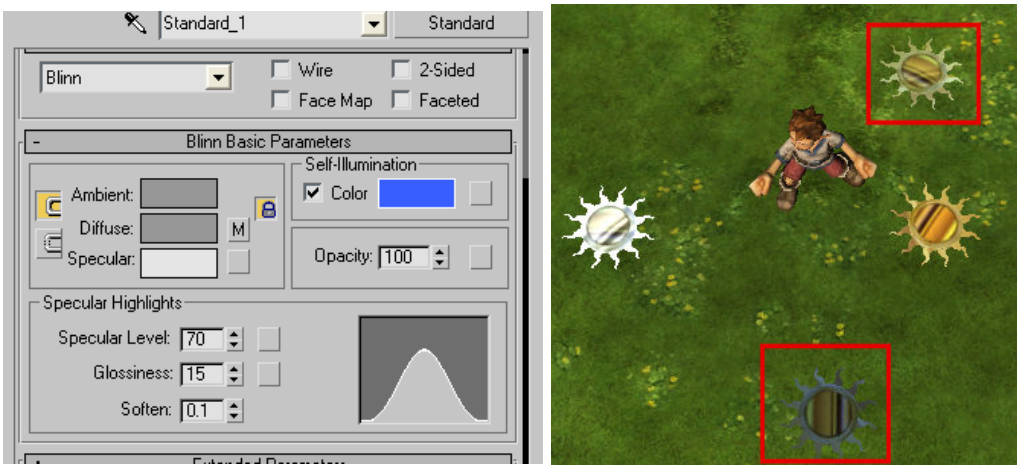
Problems with the rendering of MDL models in the game

MDLs are used for non-animated objects such as equipment and scenery props. Once you have a mesh ready in 3ds max you can export that as an .mdl file ready for use if the game with the appropriate coding has been created in the items/items.dat file.

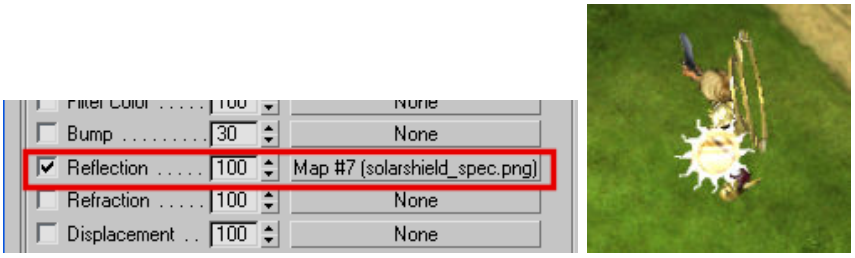
It may be the case that you've seemingly done everything right in 3ds max and the coding however the object you've made appears to have an unusual shade when it is dropped in the scene.

This is a factor related to self illumination and can easily be solved by changing the colour in the **Blinn Basic Parameters** inside the **Material Editor** window. As you can see below, the colour chosen has an influence on what colour the object is shaded when not being illuminated by an intense source of light. Note the **Yellow** and **Blue** hues respectively.



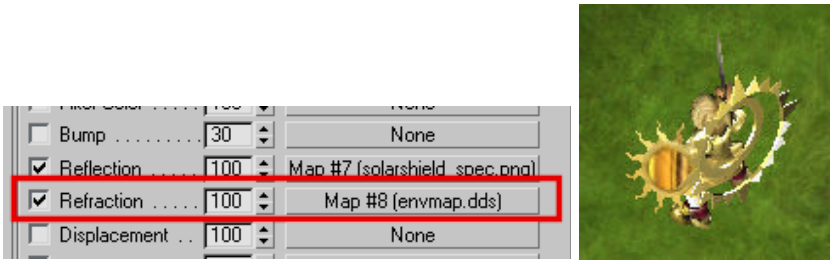


Likewise, you will eventually experience the opposite problem. The reflection map will make the object to appear to shine unnaturally and unlike the rest of other objects in the rest of the scenery as shown below.

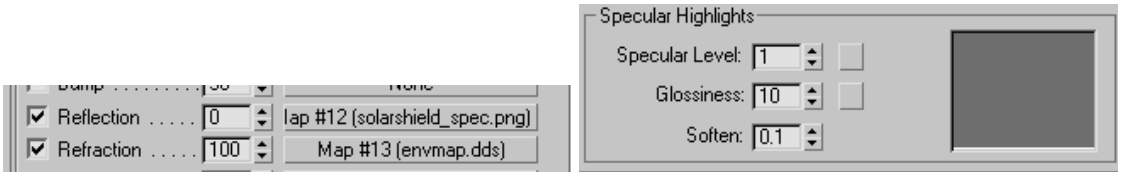


For the reflection to work decently, make sure that you set it to a very low value, such as 1 or 0 so that you will not get as much reflection.

According to Travis Baldree's guide on exportation, you also need a refraction map.
The file you need is named **envmap.dds** and you need not create one of your own since this is already present in your FATE /cubemaps folder. Like any other map, link it as a refraction map in the **Material Editor** for your object.



You may set the refraction values to be high, such as 100 to further reduce the reflection to acceptable levels. You may also need to reduce the specular highlights level to the minimum.



Include the envmap.dds file in your items folder together with the texture maps if you want your reflections to be animated, or exclude it if you want it to be static.

FURTHER NOTES 2 – ANIMATING

BONES:

The Character's bones must be either attached to the mesh via either the Physique or Skin modifier, not both at once, otherwise you will have problems with the field of selection of the character. Your character will remain highlit wherever your mouse pointer is on the screen and will make the game impossible to play.

The size of the selection field for each character is also determined by the original size of the model's skeleton. So if in 3ds max you make a large skeleton, eventhough you reduce its size in the monsters.dat scale, you will still get a monster with a huge field of selection. You can however reduce the scale of your model at the exportation stage.
As reference you may bear in mind that the size of the main character's model occupies about one or maybe two squares of the 3ds max grid.

You may always rescale the skeleton at a later stage by entering in "FIGURE" mode in the biped's tab but this may cause problems with the animation and the mesh itself.

TEXTURE:

The texture map may be assigned to the mesh object through the map editor. It may be modified through the "Unwrap UV" modifier later on.

SKIN:

Assign skin (i.e. attach the mesh to the bones) via the "PHYSIQUE" modifier not the "SKIN" one. You might want to assign the individual vertex groups manually; you may do that through "VERTEX LINKING" in "PSYSIQUE" skinning.

ANIMATING:

Once your skeleton is ready the animations may be made by turning on the "AUTOKEY".
For the "Autokey" to work you must first have assigned a starting position to the bone you would like to animate, then you may scroll the timeline to set other keypoints of the animation.

If you need to move the centre of gravity you will have to assign it a "Trajectory". Save the individual animations in SMA format, finally save the SMS file.

MAIN ANIMATIONS:

In general the main animations for each character are:

IDLE
WALK
RUN
ATTACK
DIE
CAST

You may avoid making either, although you need at least one SMA animation for the sms to load

FURTHER NOTES 3 – CREATING ARMORS

The concept behind making armors is pretty much like making monsters – you need the skinned versions (the versions with the skin modifier applied to the mesh) of the banded mail and banded boots as provided in Travis Baldree's MODKIT.

The bandedboots file is however not skinned and you will either need to do this yourself, or download the ready-skinned file I made.

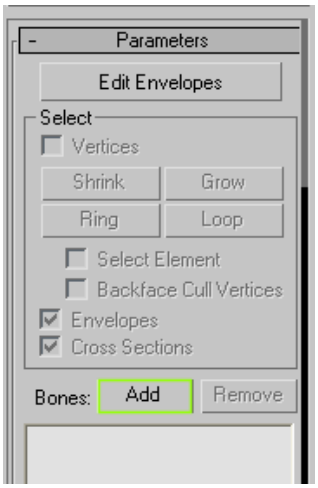
Once you have the meshes ready on your 3ds max viewport you can modify them by selecting the **Edit Mesh** submenu/modifier and moving the verteces. Otherwise you can add further pieces to the armor as additional objects and then attach them to the respective bone though the **Skin** modifier.

We will consider the Banded Mail Mesh as an example:

The skeleton of the mesh will probably have an asymmetrical pose, making it difficult for you to modify your mesh symmetrically.

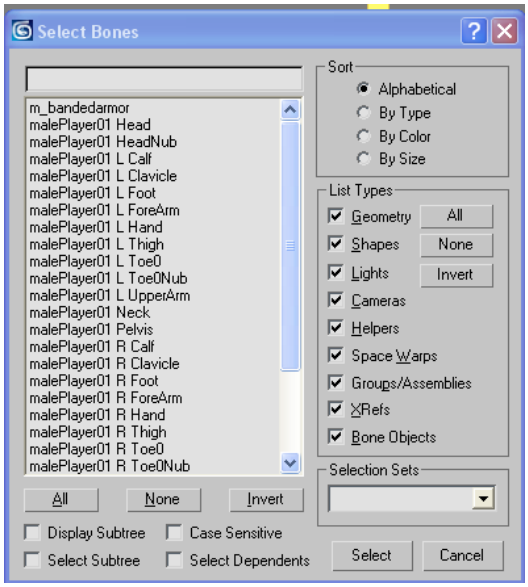
You can solve this by selecting the skeleton's **figure mode** in the **motion** panel so that it will assume the shape it had before the Skin modifier was applied.

To add new parts to your armor:
Select the new mesh object and assign a **Skin** modifier to it (make sure that your object does is in the topmost menu if you have an Edit Mesh modifier applied). In the **Parameters** panel click the **Add Bones** button and in the popup window select only the bones your object is concerned with.



For instance, if you have added an arm plate to the left arm of your character's armor, select only the **MalePlayer01 L UpperArm** bone. This way tou will avoid having to deal with the other bones taking undesired control over your new armor-component. If you need to assign more bones to a single object, you can assign the degree of influence of each bone on any vertex of your object through changing the given bone's **Abs. Effect** under the **Weight Properties** in the Skin Modifier panel, where a value of 1 stands for total control over the selected verteces.

Note that keeping the **Normalise** option checked will tell 3ds max to make bone control values for any given vertex to add up to 1 automatically.



When your armor is complete, export your file as an SMS object. In your mod directory make sure you include a \PLAYER subfolder containing a copy of the SMA and ANIMATION.DAT files of the original PLAYER folder main character so that the game will know which animations to use when your character will equip the new armor. The process of adding the armor to the game is similar to the one used for adding any other item in the game.