

Softimage Xsi Version 2.0.1 Rigging Tutorial - Beginner to intermediate.
Written by James Roberts March 2007

To create this leg rig, which is ideal for both biped and quadruped characters, simply follow these instructions.

1. Create the bones in the usual way. Using the right view port and working from top to bottom, Create> Skeleton 2d chain and name them accordingly. Double click on the node in the schematic view and type in the names; LftLegRoot, LftThighBone, LftShinBone and LftLegEff. Use a two bone chain for the leg and two one bone chains for the heel and toe. Position constrain the chain roots to the effectors above and parent effectors to the chain roots below. For example, select LftHeelRoot, then Constrain> Position , select LftLegEff. To parent everything into a complete hierarchy select LftLegEff, select Parent and the pick LftHeelRoot. Repeat this process down the hierarchy. Move the leg bones off centre at this point before you use Create> Skeleton> Duplicate symmetry in **xy** plane to create the right leg and rename the bones. The same can be done for the control objects hierarchy. Duplicate these when they are in position on the first leg.

2. Create and position the control objects. Create three square control objects and position them accordingly. Spline squares work best as you can re-orientate their centres if necessary and you can not do this with implicit squares. Create> Curve> Draw Linear and make the first one about two units by two. When created you will need to rotate the second and third squares 45 degrees in the **X** axis. With the first square control object selected, select Transform> Match Translation and pick the LftHeelEff. This will allow you to align the objects with the centre of the foot. Create three nulls. Get> Primitive> Null. Name them LftHeelNull, LftToeNull and LftKNeeNull. Position them using Transform> Match Translation aligning the LftHeelNull with the LftHeelRt and the LftToeNull with the LftToeEff. LftKNeeNull should be placed a couple of units in front of the thigh/knee bone area.

3. After naming and aligning these control objects and nulls and parent them into the hierarchy as follows; LftFootControl parents the LftHeelControl and LftToeControl. These in turn parent the nulls. LftHeelControl parents LftHeelNull and LftToeControl parents LftToeNull. See the screen grab for a full view of the hierarchy and it's constraints.

To re-cap so far, the basic principles are as follows;

4. Create, name and position the objects, be they bones, nulls or control objects, whatever their final function. Next parent into a hierarchy and begin the necessary constraints for both position and orientation in the correct order.

5. Position constraint LftLegEff to LftHeelNull.

6. Position constraint LftHeelEff to LftHeelControl.

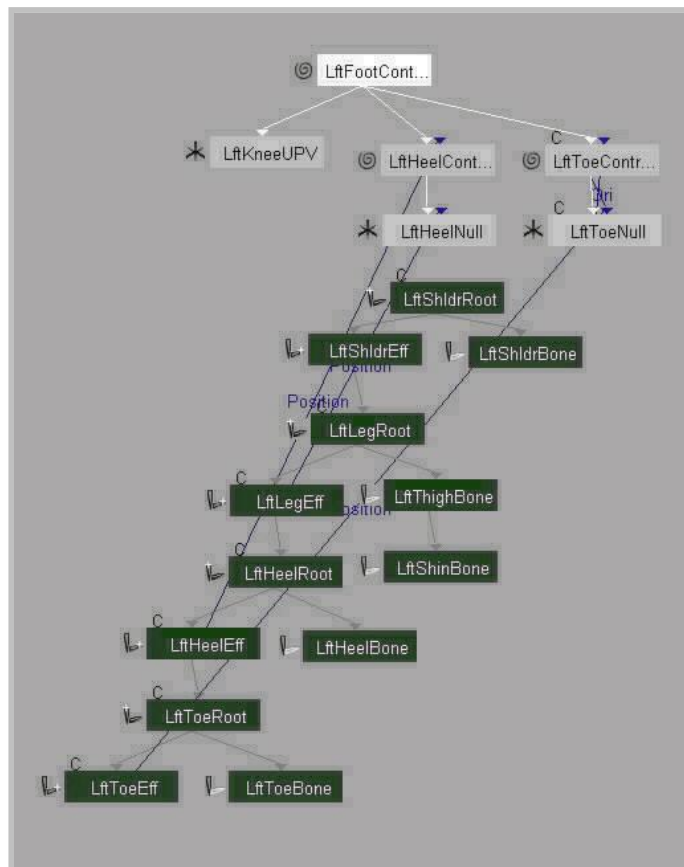
7. Position constraint LftToeEff to LftToeNull.

8. Orientation constraints must be set up in the following order; LftToeNull to LftToeControl and then LftToeControl to LftToeNull. (In fact when I experimented with this rig I found these constraints were not necessary - I would love to hear from any animators or riggers with their views and experiences on this.)

9. Finally set up an up vector constraint on the leg. Select LftKneeUPV, which is the third null, then select Skeleton> chain up-vector and pick the thigh bone. This bone may flip around, if it does don't worry. You need to do the following anyway, select bone hit enter to bring up it's property pages. Find the kinematic joint > Resolution plane and type in the value 180 degrees into the Roll slider and exit.

10. What these controls are for;

LftFootControl is for overall **x**, **y** and **z** translation and rotation.



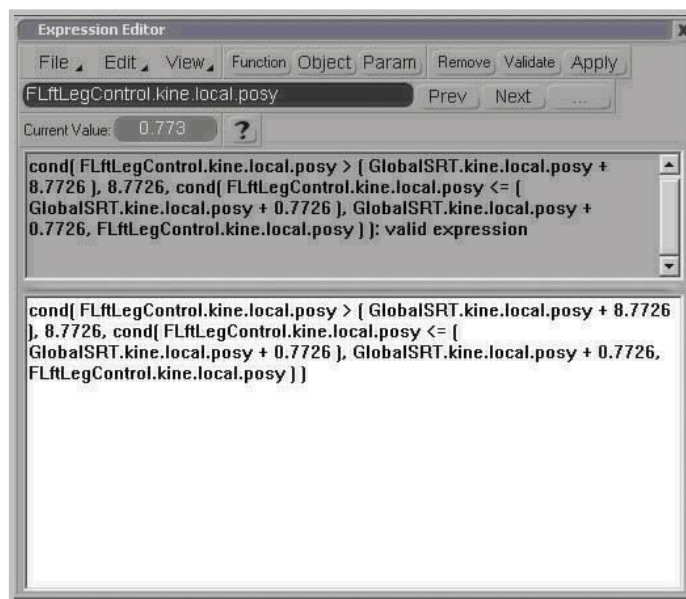
LftHeelControl is for rotation of the heel bone but does not affect the toe or leg

positions for rotations.
LftToeControl is for toe rotations.

LftKneeUPV controls any flipping around on the leg when translating the leg's position for animation.

Foot note - this is essentially just my version of a rig tutorial that is available on another web site - but I found that I could get it to work in exactly the same way with a grand total of eight less constraints and there fore a lot less work and headaches. The original has position constraints going back and forth between the LftFootControl and the two controls parented below it and orientation constraints between the nulls to the controls above and back again. I removed these and found it did not make any difference at all, at least not on my rig. Why that is - I don't know - but if works don't knock it!

Update:- Why not try this method of adding an expression between the feet controls and the top null in the hierarchy. Select the LftFootControl and then in the explorer and open up it's Kinematics> Local Transform> Pos **Y** and right click to open it's expression editor. Copy and paste this expression into the blank area,



`Cond(FLftLegControl.kine.local.posy > (GlobalSRT.kine.local.posy + 8.7726), 8.7726, cond(FLftLegControl.kine.local.posy <= (GlobalSRT.kine.local.posy + 0.7726), GlobalSRT.kine.local.posy + 0.7726, FLftLegControl.kine.local.posy))`

Click validate and Apply. Be careful to adjust your object names accordingly and values in **Y** as well. Now test the FLftLegControl and you will find that the leg will not go through the ground. But you may find that it will not reset in the **Y** axis if you create reset positions, **X** and **Z** will however.

End of Tutorial.