Hello X3D Authors 10-step process

1. **Pick target.** The target node is a Transform, and the target field is *set_rotation*.
2. **Name target.** The Transform is named DEF='EarthCoordinateSystem'.
3. **Check accessType and data type.** As shown by the Transform node field-definition table in Chapter 3 and the X3D-Edit tooltip, the *set_rotation* field has type SFRotation.
4. **Determine whether Sequencer or Script.** These special node types are not applicable to this example, because the data type for *set_rotation* is SFRotation which is a floating-point type.
5. **Determine which Interpolator.** The animating OrientationInterpolator is named DEF='SpinThoseThings' and placed just before the Transform.
6. **Triggering sensor.** A triggering TouchSensor is added next to the geometry to be clicked, and then named DEF='ClickTriggerTouchSensor'.
7. **TimeSensor clock.** The TimeSensor is added at the beginning of the chain, named DEF='OrbitalTimeInterval' and has both the cycleInterval and loop fields set.
8. **Connect trigger.** Add ROUTE to connect the triggering TouchSensor node's touchTime output field to the clock node's startTime input field.
9. **Connect clock.** Add ROUTE to connect the clock node's fraction_changed output field to the interpolator node's set_fraction input field.
10. **Connect animation output.** Add ROUTE to connect the interpolator node's value_changed output field to the original target input field, *set_rotation*.

*X3D for Web Authors*, Figure 7.5, pp. 193-195.

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter07-EventAnimationInterpolation/HelloX3dAuthorsAnimationChain.x3d

10-step process for constructing animation chains, applied to animated HelloWorld example

I strongly recommend you print this out (or keep it handy) and check off each step as you proceed. After a few times you will find that you are doing this without needing the checklist. Keeping a consistent pattern lets you avoid thinking that the various animation nodes are “really different” when they are not. It also helps you avoid skipping steps and making mistakes that are hard to debug afterwards.