**From Last Week: Task 3 - Autoworm - Model Solution**

resetMaxFile(#noPrompt)

cyl = Cylinder smooth:on sides:18 height:50 heightsegs:5 radius:5 \

transform:(matrix3 [0,1,0] [0,0,-1] [1,0,0] [0,0,0])

B1 = BoneSys.createBone [0,0,0] [10,0,0] [0,0,1]

B2 = BoneSys.createBone [10,0,0] [20,0,0] [0,0,1]

B3 = BoneSys.createBone [20,0,0] [30,0,0] [0,0,1]

B4 = BoneSys.createBone [30,0,0] [40,0,0] [0,0,1]

B5 = BoneSys.createBone [40,0,0] [50,0,0] [0,0,1]

B5.parent = B4

B4.parent = B3

B3.parent = B2

B2.parent = B1

skn = skin()

addmodifier cyl skn

max modify mode

modPanel.setCurrentObject skn

skinOps.addBone skn B1 1

skinOps.addBone skn B2 1

skinOps.addBone skn B3 1

skinOps.addBone skn B4 1

skinOps.addBone skn B5 1

animate on (

at time 10 (

rotate B2 (angleaxis -45 [0,1,0])

rotate B3 (angleaxis 45 [0,1,0])

rotate B4 (angleaxis 45 [0,1,0])

rotate B5 (angleaxis -45 [0,1,0])

)

at time 20 (

rotate B2 (angleaxis 45 [0,1,0])

rotate B3 (angleaxis -45 [0,1,0])

rotate B4 (angleaxis -45 [0,1,0])

rotate B5 (angleaxis 45 [0,1,0])

)

)

**Here is a solution for the ten-box task from last week:**

resetMaxFile()

for k in 0.0 to 324.0 by 36.0 do (

box length:1 width:1 height:1 position:[10,0,0] \

rotation:(angleAxis k [0,0,1])

)

**And here is Dave Wortley’s basic ‘Test Dialog’:**

(<https://davewortley.wordpress.com/lessons/>)

try(DestroyDialog RL\_Test)catch()

Rollout RL\_Test "The Test Dialog"

(

button btn\_ok "Ok"

on btn\_ok pressed do

(

DestroyDialog RL\_Test

)

)

CreateDialog RL\_Test

**Task 1a**

Combine these two scripts so that the ten boxes are created when the OK button is pressed.

**Task 1b**

Drag-and-drop your script onto the max toolbar to create a new button to create the dialog.

For other GUI elements, their capabilities, and their deployment details see the “Visual MAXScript Editor…”, “MAXScript Help…”, and Dave Wortley’s lesson 3:

<https://davewortley.wordpress.com/2012/06/24/lesson-3-more-building-interfaces/>

Here is the script for creating the firework animation:

fn animTrajectory speed:10 steps:10 = (

speed = float(speed)

az = -speed/(steps-2)\*2

dz = speed

newpos = $.pos

step = (animationRange.end - animationRange.start) / (steps - 1)

animate on (

for keytime in animationRange.start to \

animationRange.end by step do (

at time keytime (

$.pos = newpos

)

newpos = newpos + [0,0,dz]

dz = dz + az

)

)

)

fn animExplode radius:10 = (

radius = float(radius)

convertToMesh $

meshop.explodeAllFaces $ 0

update $

animate on (

at time ((animationRange.start + animationRange.end) / 2) (

meshop.bevelFaces $ #{1..$.numFaces} 0.0000000001 0

)

at time animationRange.end (

meshop.bevelFaces $ #{1..$.numFaces} radius 0

)

)

)

**Task 2**

Construct a GUI for this tool that allows the parameters of the functions to be set (i.e. speed, steps and radius) and calls them appropriately when a button is pressed. e.g.

Finally, here is the script I showed you that uses the faces of a mesh to create a looping animation of multiple spheres that traverse the edges.

obj = $

convertToMesh obj

for faceId = 1 to obj.numFaces do

(

vertIds = getFace obj faceId

v1 = getVert obj vertIds[1]

v2 = getVert obj vertIds[2]

v3 = getVert obj vertIds[3]

s = Sphere radius:1 segs:32

s.parent = obj

animate on (

at time 0 (s.pos = v1)

at time 10 (s.pos = v2)

at time 20 (s.pos = v3)

at time 30 (s.pos = v1)

)

)

update obj

**Task 3**

Use this script to develop a tool with an appropriate GUI that:

**a)** allows the speed of the spheres to be changed

**b)** allows any object in the scene to be picked and copied instead of always using a sphere

(hint see “pickbutton”)

**c)** add a colour picker that allows the “wirecolour” of all the newly created objects to be set