

```

if (`window -exists Stretchy`) deleteUI Stretchy;
if (`windowPref -exists Stretchy`) windowPref -remove Stretchy;
window Stretchy;

    frameLayout -collapsable false -label "Welcome to the Stretchy Tool" frame1;
    separator -w 50 -style "single";
    frameLayout -collapsable true -label "Name of Joints";
        rowLayout -numberOfColumns 5 -columnWidth3 50 50 50;
            $radio1 = `radioCollection "Radio_Buttons" `;
                string $rLeg = `radioButton -label "rLeg" -offCommand
"enableCustomName" -onCommand "disableCustomName" jointName1`;
                string $rArm = `radioButton -label "rArm" -offCommand
"enableCustomName" -onCommand "disableCustomName" jointName2`;
                string $lLeg = `radioButton -label "lLeg" -offCommand
"enableCustomName" -onCommand "disableCustomName" jointName3`;
                string $lArm = `radioButton -label "lArm" -offCommand
"enableCustomName" -onCommand "disableCustomName" jointName4`;
                string $customSI = `radioButton -label "Custom" -onCommand
"enableCustomName" -offCommand "disableCustomName" customNames1`;

setParent ..;

    frameLayout -collapsable false -label "Custom Name of Joints" customJointNames1;
    columnLayout;
        string $Custom = `textField customNameField`;
        setParent ..;

columnLayout;

    button -label "Create Locators" -command "create_locators ($radio1)";

        setParent ..;

setParent ..;

```

```
setParent ..;
```

```
    rowLayout -numberOfColumns 2 -columnWidth2 50 50;
```

```
    frameLayout -collapsible true -label "Orientation of Joints";
```

```
    frameLayout -collapsible false -label "Primary Axis";
```

```
    rowLayout -numberOfColumns 3 -columnWidth3 50 50 50;
```

```
    $primaryCollection = `radioCollection "Prime_Axis";`
```

```
        $pX = `radioButton -label "X" -onCommand "primary_orientation  
($primaryCollection)" axis1`;`
```

```
        $pY = `radioButton -label "Y" -onCommand "primary_orientation  
($primaryCollection)" axis2`;`
```

```
        $pZ = `radioButton -label "Z" -onCommand "primary_orientation  
($primaryCollection)" axis3`;`
```

```
setParent ..;
```

```
    frameLayout -collapsible false -label "Secondary Axis";
```

```
    rowLayout -numberOfColumns 4 -columnWidth4 50 50 50 50;
```

```
    $SecondaryCollection = `radioCollection "Second_Axis";`
```

```
        $sX = `radioButton -label "X" -onCommand "sec_orientation  
($SecondaryCollection)" secondary1`;`
```

```
        $sY = `radioButton -label "Y" -onCommand "sec_orientation  
($SecondaryCollection)" secondary2`;`
```

```
        $sZ = `radioButton -label "Z" -onCommand "sec_orientation  
($SecondaryCollection)" secondary3`;`
```

```
        $sN = `radioButton -label "None" -onCommand "sec_orientation  
($SecondaryCollection)" secondary4`;`
```

```
setParent ..;
```

```
    frameLayout -collapsible false -label "Secondary Axis World Orientation";
```

```
    rowLayout -numberOfColumns 3 -columnWidth3 50 50 50;
```

```
    $SecondaryWorld = `radioCollection "Second_Axis_World";`
```

```
        $swX = `radioButton -label "X" -onCommand "sec_world_orientation
($SecondaryWorld)" secondaryWorld1`;
```

```
        $swY = `radioButton -label "Y" -onCommand "sec_world_orientation
($SecondaryWorld)" secondaryWorld2`;
```

```
        $swZ = `radioButton -label "Z" -onCommand "sec_world_orientation
($SecondaryWorld)" secondaryWorld3`;
```

```
    setParent ..;
```

```
    columnLayout;
```

```
        $radioWorld = `radioCollection "plus_minus_collection"`;
```

```
            string $positiveWorld = `radioButton -label "+" -onCommand "plus_minus_orientation
($radioWorld)" positiveWorld`;
```

```
            string $negativeWorld = `radioButton -label "-" -onCommand "plus_minus_orientation
($radioWorld)" negativeWorld`;
```

```
        setParent ..;
```

```
    frameLayout -collapsible false -label "Scale";
```

```
    columnLayout;
```

```
        floatField -max 1000 -min .01 -value 1 "Ctrl_Scale";
```

```
        setParent frame1;
```

```
            frameLayout -collapsible false -label "Please Make Sure The Four Locators Are Selected"
frame1;
```

```
                columnLayout;
```

```
                    button -label "Start" -command "joint_creations ($prefix, $radio1, $PAxis,
$SAxis, $SAxisWorld, $PMAxis)";
```

```
                    radioCollection -edit -select $customSI $radio1;
```

```
showWindow Stretchy;
```

```
global proc enableCustomName() {
```

```
    textField -e -enable true customNameField;
```

```
}
```

```
global proc disableCustomName() {
```

```
    textField -e -enable false customNameField;
```

```
}
```

```
proc string get_radio_states (string $radio_names) {
```

```
    string $radio_selection = `radioCollection -q -select $radio_names`;
```

```
    return `radioButton -q -label $radio_selection`;
```

```
}
```

```
////////////////////////////////
```

```
/*Locator Creation*/
```

```
////////////////////////////////
```

```
proc create_locators (string $radiocollection) {
```

```
    string $prefix_name = `get_radio_states($radiocollection)`;
```

```
    global string $prefix[];
```

```
    if ($prefix_name == "lArm")
```

```
    {
```

```
        $prefix = {"l_shoulder_", "l_elbow_", "l_wrist_", "l_palm_"};
```

```
        for ($l = 0; $l < 4; $l++)
```

```
        {
```

```
            spaceLocator -n ($prefix[$l] + "loc");
```

```
            move $l 0 0;
```

```

        CenterPivot;
    }
}
else if ($prefix_name == "rArm")
{
    $prefix = {"r_shoulder_", "r_elbow_", "r_wrist_", "r_palm_"};
    for ($i = 0; $i <4; $i++)
    {
        spaceLocator -n ($prefix[$i] + "loc");
        move $i 0 0;
        CenterPivot;
    }
}
else if ($prefix_name == "lLeg")
{
    $prefix = {"l_hip_", "l_knee_", "l_ankle_", "l_ball_"};
    for ($i = 0; $i <4; $i++)
    {
        spaceLocator -n ($prefix[$i] + "loc");
        move $i 0 0;
        CenterPivot;
    }
}
else if ($prefix_name == "rLeg")
{
    $prefix = {"r_hip_", "r_knee_", "r_ankle_", "r_ball_"};
    for ($i = 0; $i <4; $i++)
    {
        spaceLocator -n ($prefix[$i] + "loc");

```

```

        move $l 0 0;
        CenterPivot;
    }
}
else if ($prefix_name == "Custom")
{
    string $custom_name = `textField -q -tx customNameField`;
    $prefix = {($custom_name + "_1_"),($custom_name + "_2_"),($custom_name +
"_3_"),($custom_name + "_4_")};
    for ($l = 0; $l <4; $l++)
    {
        spaceLocator -n ($prefix[$l] + "loc");
        move $l 0 0;
        CenterPivot;
    }
}
}
////////////////////
/*Orientation Procs*/
////////////////////
proc string get_orientation_states (string $orientation_names) {
    string $orientation_selection = `radioCollection -q -select $orientation_names`;
    return `radioButton -q -label $orientation_selection`;
}

proc primary_orientation (string $orientationCollectionP){
    string $orientation_primary = `get_orientation_states($orientationCollectionP)`;
    global string $PAxis;
    if ($orientation_primary == "X")

```

```

{
    $PAxis = "x";
}
else if ($orientation_primary == "Y")
{
    $PAxis = "y";
}
else if ($orientation_primary == "Z")
{
    $PAxis = "z";
}
}

proc sec_orientation (string $orientationCollectionS){
    string $orientation_secondary = `get_orientation_states($orientationCollectionS)`;

    global string $$Axis;
    if ($orientation_secondary == "X")
    {
        $$Axis = "x";
    }
    else if ($orientation_secondary == "Y")
    {
        $$Axis = "y";
    }
    else if ($orientation_secondary == "Z")
    {
        $$Axis = "z";
    }
}

```

```

proc sec_world_orientation (string $OrientationCollectionSW){
    string $orientation_sec_world = `get_orientation_states($OrientationCollectionSW)`;
    global string $$AxisWorld;
    if ($orientation_sec_world == "X")
    {
        $$AxisWorld = "x";
    }
    else if ($orientation_sec_world == "Y")
    {
        $$AxisWorld = "y";
    }
    else if ($orientation_sec_world == "Z")
    {
        $$AxisWorld = "z";
    }
    else if ($orientation_sec_world == "None")
    {
        $$AxisWorld = " ";
    }
}

```

```

proc plus_minus_orientation (string $Plus_minus){
    string $orientation_plus_minus = `get_orientation_states($Plus_minus)`;
    global string $PMAxis;
    if ($orientation_plus_minus == "+")
    {
        $PMAxis = "up";
    }
    else if ($orientation_plus_minus == "-")
    {

```

```

    $PMAxis = "down";
}
}

//////////

/*diamond Creation*/

//////////

proc diamond_creation(){

    curve -d 1 -p 0 0.698893 0.008082 -p 0.48388 -9.90663e-05 -0.001829 -p 0 -0.707107 0.003326 -p -
0.48388 -0.000177231 -0.010382 -p 0 0.698893 0.008082 -p 0 0.00817894 0.244987 -p -0.48388 -
0.000177231 -0.010382 -p 0 -0.00901003 -0.236412 -p 0.48388 -9.90663e-05 -0.001829 -p 0 0.00817894
0.244987 -p 0 -0.707107 0.003326 -p 0 -0.00901003 -0.236412 -p 0 0.698893 0.008082 -k 0 -k 1 -k 2 -k 3
-k 4 -k 5 -k 6 -k 7 -k 8 -k 9 -k 10 -k 11 -k 12 ;

    rename diamond_crv1;

}

//////////

/*LockHide Creation*/

//////////

proc lockHide (string $name, string $attribute)

{

    setAttr -lock true -keyable false -channelBox false ($name + "." + $attribute + "x");
    setAttr -lock true -keyable false -channelBox false ($name + "." + $attribute + "y");
    setAttr -lock true -keyable false -channelBox false ($name + "." + $attribute + "z");

}

//////////

/*Joint Creation*/

//////////

proc joint_creations (string $starting_names[], string $Radio_Buttons, string $Axis1, string $Axis2, string
$Axis3, string $Axis4){

    //////////

    /// User-Controlled Variables ///

```

```
////////////////////////////////////
string $startLocs[] = `ls -sl`;
string $name1 = $starting_names[0];
string $name2 = $starting_names[1];
string $name3 = $starting_names[2];
string $name4 = $starting_names[3];
string $nameArray[] = {$name1, $name2, $name3, $name4}; /* array containing names */
float $ctrlScale = `floatField -query -value "Ctrl_Scale"`;

string $prefix_name = `get_radio_states($Radio_Buttons)`;
if ($prefix_name == "rLeg")
{
    spaceLocator -name "zero";
    spaceLocator -name "one";
    select -add "zero";
}
else if ($prefix_name == "lLeg")
{
    spaceLocator -name "zero";
    spaceLocator -name "one";
    select -add "zero";
}
else if ($prefix_name == "rArm")
{
    spaceLocator -name "zero";
}
else if ($prefix_name == "lArm")
{
    spaceLocator -name "zero";
}
```

```

    }
else if ($prefix_name == "Custom")
    {
        spaceLocator -name "zero";
    }
string $valueLoc[] = `ls-sl`;
int $hv = `size $valueLoc`; /* 1 = horizontal controls, 2 = vertical controls */
delete;
string $pAxis = $Axis1;
string $sAxis = $Axis2;
string $sAxisWorld = $Axis3;
string $plusMinus = $Axis4;
string $sAxisWorldOr = ($sAxisWorld + $plusMinus);
//////////
//// Joint Creation ////
//////////
select -cl;
if (`size $startLocs` < 4)
{
    error "Please Make/Select 4 Locators";
}
if (`size $startLocs` > 4)
{
    error "Too Many Locators Selected";
}
else
{
    for ($i = 0; $i < 4; $i = $i + 1)
    {

```

```

joint -name ("bind_" + ($nameArray[$i]) + "jnt");
select $startLocs[$i];
select -add ("bind_" + ($nameArray[$i]) + "jnt");
pointConstraint -offset 0 0 0 -weight 1;
select -r ("bind_" + ($nameArray[$i]) + "jnt_pointConstraint1");
doDelete;
}
parent ("bind_" + ($nameArray[3]) + "jnt") ("bind_" + ($nameArray[2]) + "jnt");
parent ("bind_" + ($nameArray[2]) + "jnt") ("bind_" + ($nameArray[1]) + "jnt");
parent ("bind_" + ($nameArray[1]) + "jnt") ("bind_" + ($nameArray[0]) + "jnt");
}

select -hi ("bind_" + ($nameArray[0]) + "jnt");
string $bindJoints[] = `ls-sl`;
select $bindJoints[0];

print $pAxis ;
print "\n";

print $sAxis ;
print "\n";

print $sAxisWorld ;
print "\n";
    print $sAxisWorldOr ;
print "\n";

```

```

joint -e -oj ($pAxis + $sAxis + $sAxisWorld) -secondaryAxisOrient $sAxisWorldOr -ch -zso;
////////////////////////////////////
//// Control Orientation Set-Up ////
////////////////////////////////////

float $jointOr1[] = `getAttr ($bindJoints[0] + ".jointOrient")`;
float $jointOr2[] = `getAttr ($bindJoints[1] + ".jointOrient")`;
float $jointOr3[] = `getAttr ($bindJoints[2] + ".jointOrient")`;
float $jointOr4[] = `getAttr ($bindJoints[3] + ".jointOrient")`;

float $shoulder[] = `getAttr ($startLocs[0]+ ".translate")`;
float $elbow[] = `getAttr ($startLocs[1]+ ".translate")`;
float $wrist[] = `getAttr ($startLocs[2]+ ".translate")`;

float $orShoulder = ($jointOr1[1] + $jointOr1[2]);
float $orElbow = ($jointOr1[1] + $jointOr1[2] + $jointOr2[1] + $jointOr2[2]);
float $orWrist = ($jointOr1[1] + $jointOr1[2] + $jointOr2[1] + $jointOr2[2] + $jointOr3[1] +
$jointOr3[2]);

float $bindAttr[] = `getAttr ($bindJoints[0] + ".translate")`;
////////////////////////////////////
//// Orient Plane Creation ////
////////////////////////////////////

polyPlane -w 1 -h 1 -sx 1 -sy 1 -ax 0 1 0 -cuv 2 -ch 1 -n "Axis_Ref";
select -r Axis_Ref.vtx[1] Axis_Ref.vtx[0];

MergeToCenter;

select Axis_Ref.vtx[2];

xform -translation $shoulder[0] $shoulder[1] $shoulder[2];

select Axis_Ref.vtx[1];

xform -translation $elbow[0] ($elbow[1]) $elbow[2];

```

```

select Axis_Ref.vtx[0];
xform -translation $wrist[0] $wrist[1] $wrist[2];
////////////////////////////////////
//// Control Creation/Orientation ////
////////////////////////////////////
circle -name ("fk_" + $nameArray[0] + "ctrl");
xform -scale $ctrlScale $ctrlScale $ctrlScale;
xform -translation $shoulder[0] $shoulder[1] $shoulder[2];
select "Axis_Ref";
select -add ("fk_" + $nameArray[0] + "ctrl");

normalConstraint -weight 1 -aimVector 1 0 0 -upVector 1 0 0 -worldUpType "vector" -worldUpVector
0 1 0;
select ("fk_" + $nameArray[0] + "ctrl_normalConstraint1");
doDelete;
select ("fk_" + $nameArray[0] + "ctrl");

duplicate -rr -name ("fk_" + $nameArray[1] + "ctrl");
xform -translation $elbow[0] $elbow[1] $elbow[2];

duplicate -name ("fk_" + $nameArray[2] + "ctrl");
xform -translation $wrist[0] $wrist[1] $wrist[2];

if ($hv == 1) /* if the input is set to horizontal */
{
    /* X axis*/
    if (`abs($bindAttr[0])` > `abs($bindAttr[2])`)
    {
        select ("fk_" + $nameArray[0] + "ctrl");
        xform -r -os -rotation ($orShoulder * -1) 0 0;
    }
}

```

```

duplicate -rr -name ("ik_" + $nameArray[0] + "ctrl");

select ("fk_" + $nameArray[1] + "ctrl");
xform -r -os -rotation (($orElbow * -1)) 0 0;

select ("fk_" + $nameArray[2] + "ctrl");
xform -r -os -rotation (($orWrist * -1)) 0 0;
duplicate -rr -name ("ik_" + $nameArray[2] + "ctrl");
}

/* Z axis */
if (`abs($bindAttr[2])` > `abs($bindAttr[0])`)
{
select ("fk_" + $nameArray[0] + "ctrl");
xform -r -os -rotation $orShoulder 0 0;
duplicate -rr -name ("ik_" + $nameArray[0] + "ctrl");

select ("fk_" + $nameArray[1] + "ctrl");
xform -r -os -rotation ($orElbow + 90) 0 0;

select ("fk_" + $nameArray[2] + "ctrl");
xform -r -os -rotation ($orWrist + 90) 0 0;
duplicate -rr -name ("ik_" + $nameArray[2] + "ctrl");
}
}
else /* if the input is set to vertical */
{
/* X axis*/
if (`abs($bindAttr[0])` > `abs($bindAttr[2])`)
{

```

```

select ("fk_" + $nameArray[0] + "ctrl");
setAttr ("fk_" + $nameArray[0] + "ctrl.rotateX") ($orShoulder);
duplicate -rr -name ("ik_" + $nameArray[0] + "ctrl");

select ("fk_" + $nameArray[1] + "ctrl");
setAttr ("fk_" + $nameArray[1] + "ctrl.rotateX") ($orShoulder + abs($jointOr2[1]) +
abs($jointOr2[2]));

select ("fk_" + $nameArray[2] + "ctrl");
setAttr ("fk_" + $nameArray[2] + "ctrl.rotateX") (($orShoulder + abs($jointOr2[1]) +
abs($jointOr2[2]) - abs($jointOr3[1]) - abs($jointOr3[2])));
duplicate -rr -name ("ik_" + $nameArray[2] + "ctrl");
}

/* Z axis */
if (`abs($bindAttr[2])` > `abs($bindAttr[0])`)
{
select ("fk_" + $nameArray[0] + "ctrl");
setAttr ("fk_" + $nameArray[0] + "ctrl.rotateZ") ($orShoulder);
duplicate -rr -name ("ik_" + $nameArray[0] + "ctrl");

select ("fk_" + $nameArray[1] + "ctrl");
setAttr ("fk_" + $nameArray[1] + "ctrl.rotateZ") ($orElbow);

select ("fk_" + $nameArray[2] + "ctrl");
setAttr ("fk_" + $nameArray[2] + "ctrl.rotateZ") ($orWrist);
duplicate -rr -name ("ik_" + $nameArray[2] + "ctrl");
}
}

```

```

select ("fk_" + $nameArray[0] + "ctrl");
select -add ("fk_" + $nameArray[1] + "ctrl");
select -add ("fk_" + $nameArray[2] + "ctrl");
select -add ("ik_" + $nameArray[0] + "ctrl");
select -add ("ik_" + $nameArray[2] + "ctrl");
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0 -pn 1;
////////////////////////////////////
//// IK FK Switch and Elbow Twist ////
////////////////////////////////////
circle -name "ik_fk_switch";
xform -scale ($ctrlScale * .3)($ctrlScale * .3)($ctrlScale * .3);
xform -translation $wrist[0] $wrist[1] $wrist[2];
select "Axis_Ref";
select -add "ik_fk_switch";

normalConstraint -weight 1 -aimVector 1 0 0 -upVector 0 1 0 -worldUpType "vector" -worldUpVector
0 1 0;

select "ik_fk_switch";
xform -r -os -translation 0 3 0;
move -ws $wrist[0] $wrist[1] $wrist[2] ik_fk_switch.scalePivot ik_fk_switch.rotatePivot;

addAttr -ln "IK_Weight" -at "float" -min 0 -max 1 -k yes;

select ik_fk_switch_normalConstraint1;

doDelete;

select "ik_fk_switch";

makeIdentity -apply true -t 1 -r 1 -s 1 -n 0 -pn 1;

lockHide("ik_fk_switch", "t");
lockHide("ik_fk_switch", "r");
lockHide("ik_fk_switch", "s");

doGroup 0 1 1;

```

```

rename "Ik_Fk_Switch_grp";

diamond_creation;

rename "elbow_twist_ctrl";

xform -translation $elbow[0] ($elbow[1]) $elbow[2];
xform -scale ($ctrlScale * .7) ($ctrlScale * .7) ($ctrlScale * .7);
select "Axis_Ref";
select -add "elbow_twist_ctrl";

normalConstraint -weight 1 -aimVector 1 0 0 -upVector 0 1 0 -worldUpType "vector" -worldUpVector
0 1 0;

select "elbow_twist_ctrl";
xform -r -os -translation 0 2 0;

move -ws $elbow[0] $elbow[1] $elbow[2] elbow_twist_ctrl.scalePivot elbow_twist_ctrl.rotatePivot;

select elbow_twist_ctrl_normalConstraint1;

doDelete;

select "elbow_twist_ctrl";

makeIdentity -apply true -t 1 -r 1 -s 1 -n 0 -pn 1;

select "Axis_Ref";

doDelete;

////////////////////////////////////
//// Control Strings ////
////////////////////////////////////

select ("fk_" + $nameArray[0] + "ctrl");
select -add ("fk_" + $nameArray[1] + "ctrl");
select -add ("fk_" + $nameArray[2] + "ctrl");
string $fkCtrls[] = `ls-sl`;

select elbow_twist_ctrl;

```

```

string $elbowTwist[] = `ls-sl`;

select ("ik_" + $nameArray[0] + "ctrl");
select -add ("ik_" + $nameArray[2] + "ctrl");
string $ikCtrls[] = `ls-sl`;

select Ik_Fk_Switch_grp;
select -add ik_fk_switch;
string $ikFkSwitch[] = `ls-sl`;
////////////////////////////////////
//// Control/Control Parenting ////
////////////////////////////////////
parent $fkCtrls[1] $fkCtrls[0];
parent $elbowTwist[0] $fkCtrls[1];
parent $fkCtrls[2] $fkCtrls[1];
parent $ikCtrls[1] $ikCtrls[0];
////////////////////////////////////
//// Joint Duplication ////
////////////////////////////////////
select -hi $bindJoints[0];
duplicate -rr -name "ik_1";
duplicate -rr -name "fk_1";
select -cl;

for ($counter = 0; $counter < 4; $counter = $counter + 1)
{
    select ("ik_" + ($counter + 1));
    rename ("ik_" + $nameArray[$counter] + "jnt");
    select -cl;
}

```

```
for ($counter = 0; $counter < 4; $counter = $counter + 1)
{
    select ("fk_" + ($counter + 1));
    rename ("fk_" + $nameArray[$counter] + "jnt");
    select -cl;
}
```

```
select -hi ("ik_" + $nameArray[0] + "jnt");
```

```
string $ikJoints[] = `ls-sl`;
```

```
select -hi ("fk_" + $nameArray[0] + "jnt");
```

```
string $fkJoints[] = `ls-sl`;
```

```
////////////////////////////////////
```

```
//// Control/Joint Parenting ////
```

```
////////////////////////////////////
```

```
select $bindJoints[2];
```

```
select -add $ikFkSwitch[0];
```

```
parentConstraint -mo -weight 1;
```

```
////////////////////////////////////
```

```
//// Group Create ////
```

```
////////////////////////////////////
```

```
select $bindJoints[0];
```

```
select -add $ikJoints[0];
```

```
select -add $fkJoints[0];
```

```
doGroup 0 1 1;
```

```
rename "Joint_grp";
```

```
select -cl;
```

```
select $fkCtrls[0];
```

```
select -add $ikCtrls[0];
```

```
select -add $ikFkSwitch[0];  
doGroup 0 1 1;  
rename "Ctrl_grp";  
select -cl;
```

```
/* Creation Code End */
```

```
//////////
```

```
/*IK Creation*/
```

```
//////////
```

```
select -r $fkCtrls[0] ;  
select -add $fkJoints[0] ;  
orientConstraint -mo -weight 1;
```

```
select -r $fkCtrls[1] ;  
select -add $fkJoints[1] ;  
orientConstraint -mo -weight 1;
```

```
select -r $fkCtrls[2] ;  
select -add $fkJoints[2] ;  
orientConstraint -mo -weight 1;
```

```
select -r $ikJoints[0] ;  
select -add $startLocs[0] ;  
pointConstraint -mo -weight 1;
```

```
select -r $ikCtrls[1] ;  
select -add $startLocs[2] ;  
pointConstraint -mo -weight 1;
```

```

select -r $ikJoints[0] ;
select -add $ikJoints[2] ;
string $RP_IK[] = `ikHandle -n ($starting_names[0] + "rp_ik")`;

select -r $ikJoints[2] ;
select -add $ikJoints[3] ;
string $SC_IK[] = `ikHandle -n ($starting_names[0] + "sc_ik") ikSCsolver`;

select -r $RP_IK[0];
select -add $SC_IK[0];
string $ikHandle[] = `ls -sl`;

select -r $elbowTwist[0] ;
select -tgl $ikHandle[0] ;
poleVectorConstraint -weight 1;

parent $ikHandle[0] $ikCtrls[1];
parent $ikHandle[1] $ikCtrls[1];

////////////////////
/*Stretchy_Creation*/
////////////////////
////////////////////
/*Creation of Distance Node*/
////////////////////
string $startLocShape[] = `listRelatives -s $startLocs`;
string $distance1 = `createNode "distanceDimShape" -n ($starting_names[0] +
"Stretchy_Distance")`;

```

```

    string $distance2 = `createNode "distanceDimShape" -n ($starting_names[0] +
"Stretchy_Distance_2");

    string $distance3 = `createNode "distanceDimShape" -n ($starting_names[0] +
"Stretchy_Distance_3");

    select -cl;

    select $distance1;

    select -add $distance2;

    select -add $distance3;

    string $distanceNode[] = `ls -sl`;

    connectAttr ($startLocShape[0] + ".worldPosition[0]") ($distanceNode[0] + ".endPoint");
    connectAttr ($startLocShape[2] + ".worldPosition[0]") ($distanceNode[0] + ".startPoint");
    connectAttr ($startLocShape[0] + ".worldPosition[0]") ($distanceNode[1] + ".endPoint");
    connectAttr ($startLocShape[1] + ".worldPosition[0]") ($distanceNode[1] + ".startPoint");
    connectAttr ($startLocShape[1] + ".worldPosition[0]") ($distanceNode[2] + ".endPoint");
    connectAttr ($startLocShape[2] + ".worldPosition[0]") ($distanceNode[2] + ".startPoint");

    //////////////////////////////////////
    /*Creation of Mul/Divid and Clamp Nodes*/
    //////////////////////////////////////

    string $Stretchy_P_M = `shadingNode -asUtility plusMinusAverage -n ($starting_names[0] +
"Stretchy_P_M1");

    connectAttr ($distanceNode[1] + ".distance") ($Stretchy_P_M + ".input1D[0]");
    connectAttr ($distanceNode[2] + ".distance") ($Stretchy_P_M + ".input1D[1]");

    string $stretchy1 = `shadingNode -asUtility multiplyDivide -n ($starting_names[0] +
"Stretchy_M_D1");

    string $stretchy2 = `shadingNode -asUtility multiplyDivide -n ($starting_names[0] +
"Stretchy_M_D2");

    connectAttr ($distanceNode[0] + ".distance") ($stretchy1 + ".input1.input1X");
    connectAttr ($ikCtrls[0] + ".sx") ($stretchy1 + ".input2.input2X");

```

```

connectAttr ($ikCtrls[0] + ".sx") ($stretchy2 + ".input2.input2X");
float $distanceNum = `getAttr ($Stretchy_P_M + ".output1D")`;
setAttr ($stretchy2 + ".input1X") $distanceNum ;

string $stretchy3 = `shadingNode -asUtility multiplyDivide -n ($starting_names[0] +
"Stretchy_M_D3")`;

setAttr ($stretchy3 + ".operation") 2;

connectAttr ($stretchy1 + ".output.outputX") ($stretchy3 + ".input1.input1X");
connectAttr ($stretchy2 + ".output.outputX") ($stretchy3 + ".input2.input2X");

string $clamp = `createNode "clamp" -n ($starting_names[0] + "Stretchy_Clamp1")`;
setAttr ($clamp + ".minR") 1;
setAttr ($clamp + ".maxR") 10000;
connectAttr ($stretchy3 + ".output.outputX") ($clamp + ".input.inputR");

int $size_IK_jnts = `size $ikJoints`;
for ($c = 0; $c < ($size_IK_jnts - 2); $c++)
{
    connectAttr ($clamp + ".output.outputR") ($ikJoints[$c] + ".sx");
}

////////////////////////////////////
/* Blend Colors Node Creation */
////////////////////////////////////

string $BC1 = `shadingNode -asUtility blendColors -n ($starting_names[0] + "scale_BC")`;
string $BC2 = `shadingNode -asUtility blendColors -n ($starting_names[0] + "rotate_BC")`;
string $BC3 = `shadingNode -asUtility blendColors -n ($starting_names[1] + "scale_BC")`;
string $BC4 = `shadingNode -asUtility blendColors -n ($starting_names[1] + "rotate_BC")`;
string $BC5 = `shadingNode -asUtility blendColors -n ($starting_names[2] + "rotate_BC")`;

select -cl;

select $BC1;

```

```
select -add $BC2;
select -add $BC3;
select -add $BC4;
select -add $BC5;
```

```
string $blend_color_nodes[] = `ls -sl`;
int $size_blendNodes = `size $blend_color_nodes`;
connectAttr ($ikJoints[0] + ".s") ($BC1 + ".color1");
connectAttr ($ikJoints[0] + ".r") ($BC2 + ".color1");
connectAttr ($ikJoints[1] + ".s") ($BC3 + ".color1");
connectAttr ($ikJoints[1] + ".r") ($BC4 + ".color1");
connectAttr ($ikJoints[2] + ".r") ($BC5 + ".color1");
connectAttr ($fkJoints[0] + ".s") ($BC1 + ".color2");
connectAttr ($fkJoints[0] + ".r") ($BC2 + ".color2");
connectAttr ($fkJoints[1] + ".s") ($BC3 + ".color2");
connectAttr ($fkJoints[1] + ".r") ($BC4 + ".color2");
connectAttr ($fkJoints[2] + ".r") ($BC5 + ".color2");
connectAttr ($ikFkSwitch[1] + ".IK_Weight") ($blend_color_nodes[0] + ".blender");
connectAttr ($ikFkSwitch[1] + ".IK_Weight") ($blend_color_nodes[1] + ".blender");
connectAttr ($ikFkSwitch[1] + ".IK_Weight") ($blend_color_nodes[2] + ".blender");
connectAttr ($ikFkSwitch[1] + ".IK_Weight") ($blend_color_nodes[3] + ".blender");
connectAttr ($ikFkSwitch[1] + ".IK_Weight") ($blend_color_nodes[4] + ".blender");
connectAttr ($BC1 + ".output") ($bindJoints[0] + ".s");
connectAttr ($BC2 + ".output") ($bindJoints[0] + ".r");
connectAttr ($BC3 + ".output") ($bindJoints[1] + ".s");
connectAttr ($BC4 + ".output") ($bindJoints[1] + ".r");
connectAttr ($BC5 + ".output") ($bindJoints[2] + ".r");
```

```
select -cl;
```

```
select $startLocs[1];  
select -add $startLocs[3];  
delete;  
}
```