Tutorials

endorphin™

dynamic motion synthesis

naturalmotion

Contents

Before you begin

The *endorphin* tutorials are a set of lessons that describe different aspects of the *endorphin* dynamic motion synthesis animation system.

Working through the tutorials

Each tutorial is self-contained, so you can work through them in any order. However, you will get the most out of the tutorials if you work through them in the numbered sequence. This is because the first time a concept it is introduced, it will be described in detail. In subsequent tutorials, the same concept will generally be treated as assumed knowledge.

Using the User Guide

The tutorials should be worked through in conjunction with the User Guide. The User Guide is the best place to find complete technical descriptions of *endorphin* functionality. In contrast, the tutorials are focused on teaching you how to use the system.

Tutorial 1

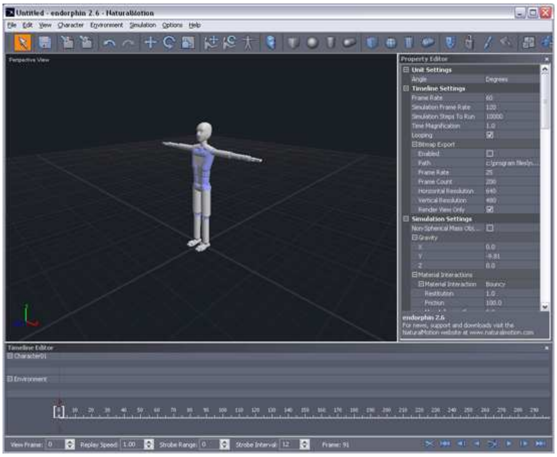
Introduction to *endorphin*

In this tutorial, we will introduce some basic *endorphin* features and tools. Starting from an empty scene, you will create animation using *endorphin's* adaptive behaviours.

Many commands have corresponding keyboard shortcuts. Whenever keyboard shortcuts are described, these are the *endorphin* default shortcuts. You can change keyboard shortcuts by selecting **Options > Keyboard** and changing these settings using the **Keyboard Options** dialog.

Step 1: Creating a new scene

In this step we will launch *endorphin* and create a new scene. Each time you create new animation using *endorphin* you will begin by creating a new scene. Scenes contain the simulated virtual world, along with all the characters involved in the simulation, including prop characters such as buildings, vehicles and weapons.


Untitled - endorphin 2.6 - NaturalMotion
File Edit View Character Environment Simulation Options Help
Perspective View
Property Editor
Timeline Editor
Character01
Environment

Unit Settings	
Angle	Degrees
Timeline Settings	
Frame Rate	60
Simulation Frame Rate	120
Simulation Steps To Run	10000
Time Magnification	1.0
Looping	☑
Bitmap Export	
Enabled	☐
Path	c:\program files\n...
Frame Rate	25
Frame Count	200
Horizontal Resolution	640
Vertical Resolution	480
Render View Only	☑
Simulation Settings	
Non-Spherical Mass Obj...	☐
Gravity	
X	0.0
Y	-9.81
Z	0.0
Material Interactions	
Material Interaction	Bouncy
Restitution	1.0
Friction	100.0

endorphin 2.6
For news, support and downloads visit the NaturalMotion website at www.naturalmotion.com

View Frame: 0 Replay Speed: 1.00 Strobe Range: 0 Strobe Interval: 12 Frame: 91

To create a new scene:

1. Run *endorphin*. Each time *endorphin* is run, it creates a new default scene. This scene contains a single *endorphin* character located at the origin in a T-pose.
2. Save the scene. Select **File > Save Scene**. The keyboard shortcut is **Ctrl+S**.

   Alternatively, click the **Save Scene** button on the Main Toolbar. Scenes are stored as binary files with a filename extension of **.ens**. Once you have saved the scene, the scene name is displayed in the System Bar.

Step 2: Working with viewports

In this step we learn how to pan, rotate and zoom in a viewport, and also how to change the viewport type.

To pan, rotate and zoom in a viewport:

The basic viewport operations are as follows:

- To **rotate** a viewport, drag the mouse with the **middle** mouse button down.
- To **pan** a viewport, drag the mouse with the **middle** and **right** buttons down.
- To **zoom** a viewport, use the **mouse wheel**.

endorphin also supports the standard Maya viewport operations:

- To **rotate** a viewport, hold down the **Alt** key and drag the mouse with the **left** mouse button down.
- To **pan** a viewport, hold down the **Alt** key and drag the mouse with the **middle** mouse button down.
- To **zoom** a viewport, hold down the **Alt** key and drag the mouse with the **right** mouse button down.

To change the viewport type:

Viewports are either a **perspective** display or an **orthogonal** display. The orthogonal display types are Front View, Back View, Left View, Right View, Top View and Bottom View. You can set the viewport type by using the following keyboard shortcuts, or by right-clicking in the viewport and selecting the viewport type from the context menu.

- Press **P** to set the viewport type to Perspective View.
- Press **F** to set the viewport type to Front View. Pressing **F** again toggles between Front View and Back View.
- Press **L** to set the viewport type to Left View. Pressing **L** again toggles between Left View and Right View.
- Press **T** to set the viewport type to Top View. Pressing **T** again toggles between Top View and Bottom View.

To display multiple viewports:

You can either display a single viewport or a grid of four viewports. The keyboard shortcut for toggling between a single viewport and multiple viewports is **V**.

Alternatively, click the **Toggle Multiple Viewports** button in the Main Toolbar.

To reset a viewport:

If you have panned, rotated or zoomed a viewport, you can reset it back to its default by selecting **View > Reset Viewport**. The keyboard shortcut is **Ctrl+R**. Alternatively, right-click in the viewport and select **Reset Viewport**.

To focus an object in a viewport:

If you have selected one or more objects in a viewport, you can focus them at the centre of the viewport by selecting **View > Focus Viewport**. The keyboard shortcut is **Ctrl+F**. Alternatively, right-click in the viewport and select **Focus Viewport**.

To rotate an orthogonal viewport:

By default, you cannot rotate any of the orthogonal viewports. To allow viewports to be rotated, select **View > Ortho Lock** and ensure it is turned off. When this setting is turned off, you can rotate orthogonal viewports.

Step 3: Working with simulations

In this step we will run a simulation to quickly generate animation, and then replay the simulation.

To run a simulation:

1. To start a simulation, click the **Simulate** button on the Simulation Toolbar. Alternatively, select **Simulation > Simulate**. The keyboard shortcut is **Ctrl+Spacebar**.
2. As the simulation runs, the character falls towards the ground, under the influence of gravity.
3. To stop a simulation, click the **Simulate** button again. Alternatively, select **Simulation > Play/Stop**. The keyboard shortcut is **Spacebar**. Note that during the simulation, the Simulate button graphic has changed to the highlighted Stop button.

To replay a simulation:

1. To replay a simulation, click the **Play** button on the Simulation Toolbar. Alternatively, select **Simulation > Play/Stop**. The keyboard shortcut is **Spacebar**.
2. To stop a replay, click the **Play** button on the Simulation Toolbar again. Alternatively, select **Simulation > Play/Stop**. The keyboard shortcut is **Spacebar**.

To navigate though animation:

- You can drag the **Time Slider** on the Timeline Editor to manually browse different frames of the animation. This is often called **scrubbing** the timeline.
- You can use the **Go To First Frame**, **Go To Previous Frame**, **Go To Next Frame** and **Go To Last Frame** buttons in the Simulation Toolbar. The corresponding keyboard shortcuts are **Ctrl+LeftArrow**, **LeftArrow**, **RightArrow** and **Ctrl+RightArrow**.

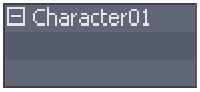
Step 4: Adding a behaviour

In this step we will add a behaviour to the character timeline. Behaviours are adaptive actions that can be applied to characters to modify their motion during a simulation.

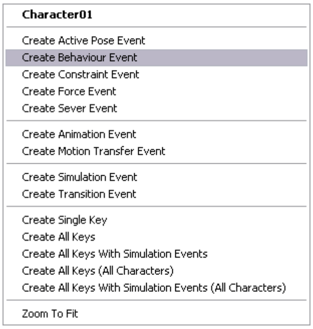
The behaviour we will add is the **Stagger** behaviour. This behaviour causes a character to react to falling motion by trying to prevent the fall.

To add a behaviour:

1. Select the character by clicking on the **Character01** zone in the Timeline Editor.

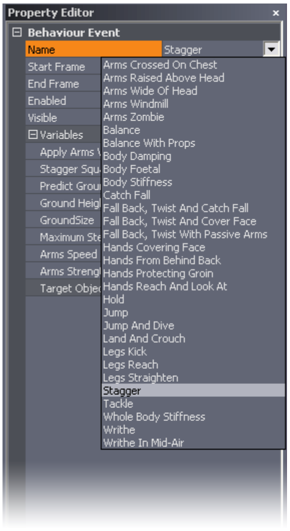


2. Add a behaviour to the character by selecting **Character > Create Behaviour Event**. Alternatively, click the **Create Behaviour Event** button in the Main Toolbar.
3. Another way to create a behaviour is to right-click on the Character01 timeline track in the Timeline Editor and select **Create Behaviour Event** from the context menu.



Once you have added the behaviour, a new **timeline marker** will appear on the Timeline Editor. This is the behaviour event marker. The orange highlight colour indicates that the event is currently selected.

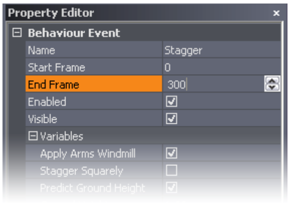
4. With the new behaviour event selected, locate its **Name** property in the Property Editor. The Name property indicates the behaviour type. Select **Stagger 3** from the dropdown list.



5. We want the behaviour event to begin at **Frame 0** and end at **Frame 300**.

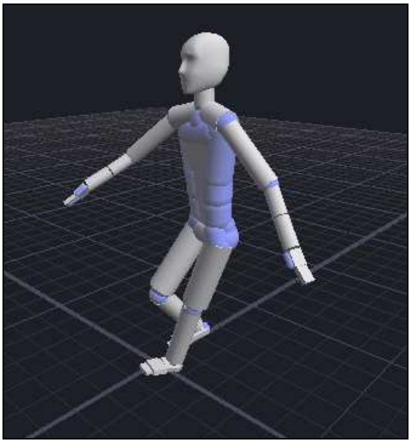
   You can set the start frame and end frame of the behaviour by editing the **Start Frame** and **End Frame** properties of the behaviour event using the Property Editor.

   You can also change the timing by dragging the marker itself in the Timeline Editor. Select and drag the left edge of the marker to set its start frame. Select and drag the right edge of the marker to set its end frame.



Behaviour Event	
Name	Stagger
Start Frame	0
End Frame	300
Enabled	✓
Visible	✓
Variables	
Apply Arms Windmill	✓
Stagger Squarely	☐
Predict Ground Height	✓

6. Test the behaviour event. Run a new simulation by clicking the **Simulate** button in the Simulation Toolbar, or by pressing **Ctrl+Spacebar**. The character should waver a bit and start to fall, and then move its legs to attempt to stop the fall, and finally catch itself with its hands and elbows. This motion is generated by the behaviour event.



You can compare the effect of the behaviour by enabling and disabling the behaviour event. To disable a timeline event, turn off its **Enabled** property using the Property Editor, or right-click on the corresponding marker in the Timeline Editor and select **Disable Event**. When an event is disabled it does influence simulations.

7. Experiment with different behaviour timings. Try changing the start frame to a different frame, such as **Frame 12**, and resimulating. The timing of the behaviour affects the overall animation. The later the behaviour begins, the more the character will have fallen. The behaviour will adaptively respond to each new situation and generate unique animation for each new starting frame. This adaptive generation of animation is the feature of *endorphin*.

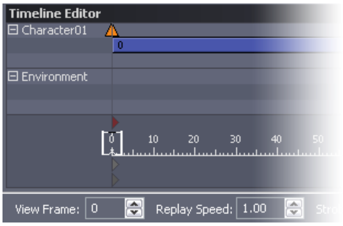
Step 5: Applying a force

In this step we will apply a **force event** to the character during the simulation. This force will destabilise the character, and change the motion generated by the Stagger behaviour.

To add a force event:

1. Move the Time Slider to **Frame 0**.
2. Create a force event by right-clicking in the Character01 timeline in the Timeline Editor and selecting **Create Force Event**. Alternatively, select Character01 in the Timeline Editor, and select **Character > Create Force Event**, or click the **Create Force Event** button in the Main Toolbar.

Once you have added the force, a new triangular **timeline marker** will appear on the Timeline Editor. This is the force event marker. The orange highlight colour indicates that the event is currently selected. In addition, a force **graphic** appears in the viewport, indicating the magnitude and direction of the force.



To edit the force target object set:

Force events apply to one or more mass objects. To edit the target object set, you need to enter **Selection** mode.

1. You can enter **Selection** mode for a force event by selecting the event, and then clicking the **[Select]** hyperlink button in the Property Editor. However, when you first create a new force event, *endorphin* automatically enters Selection mode. In Selection mode, the viewport label displays **Selecting…**, and the cursor changes to the three-dot **selection cursor**.

2. In Selection mode, click on mass objects of the character to select them as the target object. Hold down the **Ctrl** key to select multiple mass objects. If you select a collision object or graphical object, the corresponding parent mass object is selected instead. The target mass objects are displayed using a red highlight colour. The **Target Objects** property in the Property Editor displays the target mass object set.

   For this step, select the **HeadMass** mass object. The easiest way to do this is to select the **HeadGraphic** graphical object. This will automatically select the HeadMass mass object.

3. To exit Selection mode, click **Esc** or **right-click** in the viewport.

To edit the force direction and magnitude:

Force events have a direction and magnitude. You can set these using the Property Editor, or by manipulating the force graphic in the viewport. Keep in mind that when you are applying a force to multiple mass objects, you are effectively multiplying the magnitude of the force by the number of target mass objects.

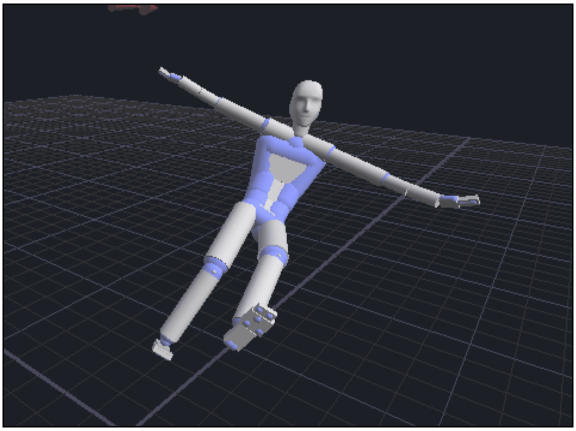
1. Select the force event marker in the Timeline Editor, or select the force graphic in the viewport.
2. Modify the **Strength** property in the Property Editor to change the magnitude of the force. Modify the **Elevation** and **Rotation** properties in the Property Editor to change the direction of the force.
3. To edit the force using the force graphic, select the force, and then select **Edit > Rotate** to activate the Rotate tool. The keyboard shortcut is **W**.

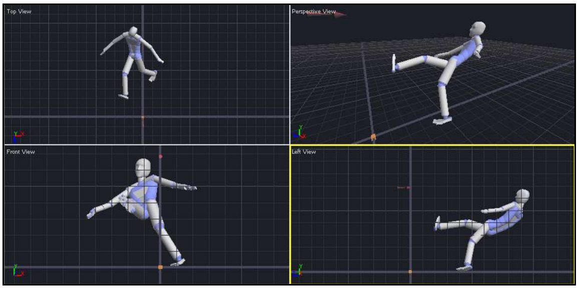
   Alternatively, click the **Rotate** button in the Main Toolbar.
4. To rotate the force direction about the **X**, **Y** or **Z** axes, click and drag the red, green or blue circular manipulators.

   To rotate the force direction in the **screen plane**, click and drag the outer light-blue circular manipulator.

   To freely rotate the force direction, click and drag the region inside the outer circular manipulator. The inner region is displayed in light grey, which indicates free rotation mode.

   To cancel a rotation during a rotation, **right-click** anywhere in the viewport.
5. To change the force magnitude, drag the purple conical manipulator at the end of the force graphic. Drag the manipulator towards the force graphic to reduce the force magnitude. Drag the manipulator away from the force graphic to increase the force magnitude.
6. For this tutorial, rotate the force so that it is directed forward towards the head mass object. Also, set the magnitude of the force to around 3.0.
7. Experiment with different force magnitudes and directions, and different timings for the force event and Stagger behaviour event. Each new combination of these parameters will adaptively generate a different animation. Using multiple viewports is a good way to examine the animation from different directions.



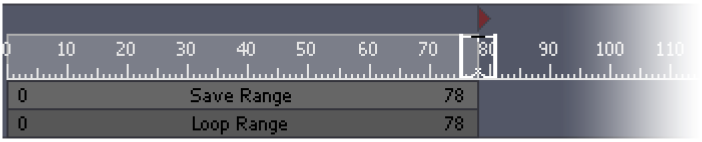


Step 6: Exporting animation

In this step, we will export the motion generated by the simulation to an external animation file. *endorphin* can export animation to FBX, XSI, BVH and Acclaim AMC animation data files, as well as Vicon V motion capture files.

To export animation:

1. Specify the range of frames that you want to export. This is the **Save Range**. The Save Range is displayed as a grey marker in the Timeline Editor.



2. Select the character that you want to use as the exported animation data source. The easiest way to select a character is to click the character in the Timeline Editor. You can also use Node View to select a character. In our example, select the Character01 timeline track.

3. Select **File > Export…**, or click the **Export** button on the Main Toolbar. The keyboard shortcut is **Ctrl+E**.

4. The **Export As** dialog is displayed. Browse for a folder to export to, and specify a new file name or select an existing file to replace it. Set the **Save As Type** to specify the export file format. Click **Save**.

5. The **Export Options** dialog is then displayed. Adjust any animation export settings as required, then click **OK** to export the animation.

   Some export formats allow the frame rate to be specified using the **Frame Rate** setting. Other export formats do not allow the Frame Rate to be set in this dialog. In these cases, *endorphin* will export using the **Frame Rate** property in the Timeline Settings. A separate tutorial will deal with animation export in greater detail.

Conclusion

In this tutorial, you have now produced animation data which can be exported into your 3D pipeline.

This tutorial has only touched on the power that *endorphin* gives you. In later tutorials, we will introduce more complex adaptive behaviours. We will pose the character and show you how to build environment objects that characters can interact with. We will also show how existing animation data can be imported, modified and exported from *endorphin.*

Tutorial 2

Using multiple behaviours

In the previous tutorial, the *endorphin* scene that we created used only a single behaviour. Although the animation looked realistic, you can create more complex motion by using multiple behaviours for a given character in a scene.

In this tutorial, you will create an animation of a person jumping from a building to a railing. You will build this animation starting with a **Jump And Dive behaviour**, and then extend the motion so that the character holds onto the railing using a **constraint event**.

Step 1: Loading the scene

In this step we will open an existing scene that contains some elements of our desired final scene.

To load the initial scene:

1. Launch *endorphin*.
2. Select **File > Open Scene…** to browse for a scene. The keyboard shortcut is **Ctrl+O**.
3. Open the scene file **Resources\Tutorials\Tutorial 2 - Using Multiple Behaviours\Tutorial 2 - Using Multiple Behaviours - Begin.ens**.

Step 2: Jumping from the platform

In this step we will add a **Jump And Dive** behaviour to the character timeline in the Timeline Editor. This behaviour will cause the character to jump forward. Our goal is to have the character jump and dive towards the railing.

When you are working on a scene, keep in mind that simulations are calculated forward from frame to frame. It is good **workflow** practise to work forward, adding timeline events necessary to create the next section of animation, and getting this section correct before moving on. Modifying the parameters of an earlier timeline event can cause a knock-on effect throughout the simulation. Even small changes to earlier timeline events can have large changes as the simulation progresses.

To add a jump behaviour to the character:

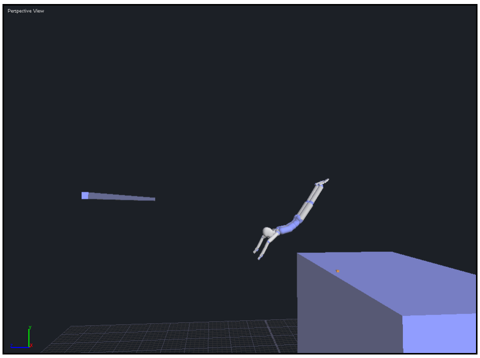
1. Add a new behaviour event to the Character01 timeline.
2. Set the behaviour type to **Jump And Dive**.
3. Adjust the timing of the behaviour so that it starts at **Frame 0** and ends around **Frame 71**. You can experiment with small changes to the start and end frames, in order to generate slightly different animations.

   For more information on adding behaviour events to characters, see **Tutorial 1: Introduction to *endorphin***.

To modify the strength of the jump behaviour:

- You can change the **Strength** property of a Jump And Dive behaviour to change the degree to which the character jumps. The default strength is 0.5. Experiment with different strength values.
- Set the jump strength to 1.0. The character now jumps further. However, if you will notice that at a strength of 1.0, the character's feet slip as the character jumps.

  Each time you change the strength or timing of the jump behaviour, resimulate the scene to test the effect of the change. For more information on running simulations, see **Tutorial 1: Introduction to *endorphin***.



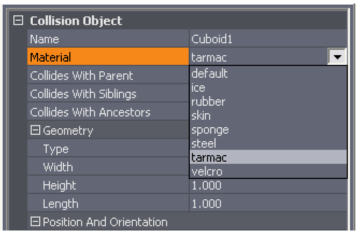
Step 3: Preventing foot slip

In this step we will modify the material of the platform that the character is standing on. By changing its material so that it has a greater degree of **friction**, you can prevent the foot slip that occurs when the character jumps.

To change the material of the platform:

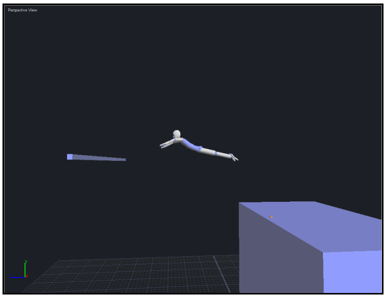
1. Select the **Cuboid01** collision object that the character is standing. You can select this object by clicking in the viewport. Alternatively, you can select this object using the **Node View**.

2. Change the **Material** property if the Cuboid01 collision object to **Tarmac**. Tarmac is a much grippier material than the default material. This will increase the frictional force between the character's feet and the platform. Keep in mind that *endorphin* is a physically-based dynamic simulator. This means that simulations are affected by physical interactions such as friction.



Collision Object	
Name	Cuboid1
Material	tarmac
Collides With Parent	default
Collides With Siblings	ice, rubber
Collides With Ancestors	skin
Geometry	sponge
Type	steel
Width	tarmac, velcro
Height	1.000
Length	1.000
Position And Orientation	

3. Resimulate the scene. You should find that the character jumps further now that its feet do not slip on the platform.

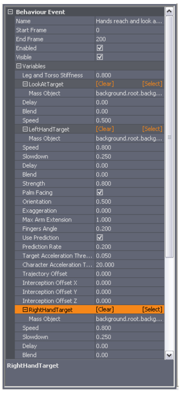


Step 4: Reaching for the railing

In this step we will add a **Hands Reach And Look At** behaviour to the character timeline in the Timeline Editor. This behaviour will cause the character to reach out towards the railing. It will also cause the character to move its head to follow the railing. Look And Reach behaviours are very useful when you want to make a character appear more lifelike.

To add a look-and-reach behaviour to the character:

1. Add a new behaviour event to the Character01 timeline. Set the behaviour type to Hands Reach And Look At, and adjust the timing of the event so that it begins at Frame 71 and ends at Frame 112.
2. You can optionally specify targets for the left hand, right hand and head. When a target has been specified for one of the hands, they will reach out to follow the target. Similarly, when a target has been specified for the head, the character will turn its neck to that it tracks the target object.
3. With the new behaviour selected, click the **[Select]** command hotlink for the **Look At Target** in the Property Editor. This places *endorphin* into **Selection** mode. Select the **Cuboid02** collision object as the look-at target. This will make the character look at the railing during the simulation. This behaviour works even when the both character and look-at target are in motion.
4. Select the same Cuboid02 collision object as the **Left Hand Target** and **Right Hand Target**. This will make the character reach out with both arms towards the railing. Each time you add a new target object, resimulate the scene. This lets you see the effect of each change you make to the scene.



Behaviour Event	
Name	Hands reach and look a...
Start Frame	0
End Frame	200
Enabled	✓
Visible	✓
Variables	
Leg and Torso Stiffness	0.800
LookAtTarget	[Clear] [Select]
Mass Object	background.root.backg...
Delay	0.00
Blend	0.00
Speed	0.500
LeftHandTarget	[Clear] [Select]
Mass Object	background.root.backg...
Speed	0.800
Slowdown	0.250
Delay	0.00
Blend	0.00
Strength	0.800
Palm Facing	✓
Orientation	0.500
Exaggeration	0.000
Max Arm Extension	1.000
Fingers Angle	0.200
Use Prediction	✓
Prediction Rate	0.200
Target Acceleration Thre...	0.050
Character Acceleration T...	20.000
Trajectory Offset	0.000
Interception Offset X	0.000
Interception Offset Y	0.000
Interception Offset Z	0.000
RightHandTarget	[Clear] [Select]
Mass Object	background.root.backg...
Speed	0.800
Slowdown	0.250
Delay	0.00
Blend	0.00

RightHandTarget

To slow down the reach:

- You can change the rate at which the character reaches for the target objects. Select the Look And Reach behaviour, and change the **Left Hand Blend** and **Right Hand Blend** properties to 0.2. If you resimulate, you will note that the reaching movement of the arms is now slower.
- You can also set different values for each arm for a slightly asymmetric reach. This can make the reaching behaviour appear more lifelike.

To adjust the reach target:

- By default, hands will reach for the **centre** of their target objects. In our case, we would like the hands to reach for slightly different locations on the target object. This will make the reaching behaviour appear more lifelike.
- Select the Look And Reach behaviour, and change the **Left Hand Interception Offset X** and **Right Hand Interception Offset X** properties to +0.3 and -0.3 respectively. These offsets are applied to the centre of the target object. By using different offsets for each hand, they now reach out in more plausible manner.
- Resimulate the scene and note the change to the simulation with the addition of these offsets. You can experiment with different offset values for each hand.

Step 5: Catching the railing

In this step we will add a **constraint event** to the character timeline. Constraint events are used to lock objects together during a simulation. By constraining the character's hands to the railing, the character effectively **catches** the railing.

To add a constraint event:

1. Add a **constraint** event to the **Character01** timeline. Right-click on the character timeline and select **Create Constraint Event**. Alternatively, select Character01 in the Timeline Editor, and then select **Character > Create Constraint Event**, or click the **Create Constraint Event** button in the Main Toolbar.
2. Adjust the Start Frame and End Frame properties of the constraint event so that it starts at Frame 109 and ends around Frame 400.

To select the constrained objects:

1. When you create a new constraint, *endorphin* automatically enters **Selection** mode. This mode allows you to select the mass objects that you want to lock together using the constraint event. To re-enter Selection mode, click the **[Select]** command hotlink in the constraint event Property Editor.

2. In Selection mode, select the **LeftFingersMass** and **RightFingersMass** mass objects. You will need to hold down the **Ctrl** key in order to select both mass objects.

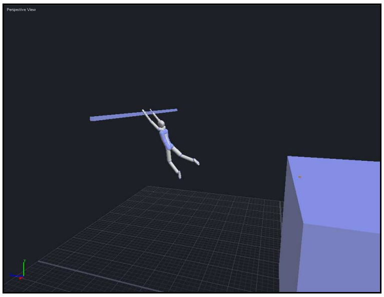
   When you are in Selection mode, you can pan, rotate and zoom the viewport. You can also change the viewport to display a different view, such as Top View or Front View, and you can also use multiple viewports. You will often need to change the view while selecting all the desired target objects.

   To exit Selection mode, press **Esc** or **right-click** in the viewport.

To set the constraint type:

1. Constraint events have different types. In this tutorial you will use the default **Lock** constraint type. This means that the constraint event will lock to the positions and orientations of the selected mass objects over the duration of the event.

2. However, we would like to relax the rotation of the hands around the X axis. This will allow the character to effectively swing around the railing. To allow rotation about the X axis, select the constraint event, and set the **Angular Strength X** property to 0.0.



3. Run the simulation again. You should find that the character now jumps, reaches for the railing and then catches it.

Step 6: Adding a leg kick

In this step we will add a **Legs Kick** behaviour to the character timeline. This behaviour will cause the character to kick out as it swings on the railing. Adding behaviours like Legs Kick is a good way to make the character appear more lifelike.

To add a leg kick behaviour to the character:

1. Add a **Legs Kick** behaviour to Character01. Adjust the timing of the behaviour so that it starts and ends approximately at **Frame 150** and **Frame 350**. Change the timing of the behaviour and simulate to see how this behaviour modifies the overall simulation.
2. You can also modify the **Strength** property of the Legs Kick behaviour. The default strength is 0.5. Experiment with different strength values, and run the simulation again to compare results.

Conclusion

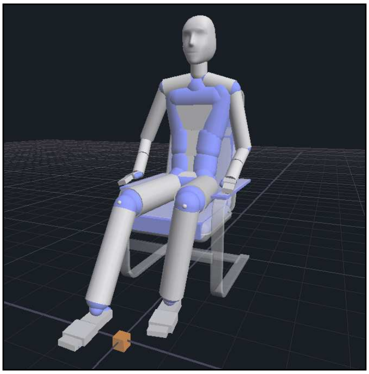
In this tutorial, you have used multiple behaviours, along with constraint events, to create more complex animation.

If you have any problems with this tutorial, open the corresponding scene **Resources\Tutorials\Tutorial 2 - Using Multiple Behaviours\Tutorial 2 - Using Multiple Behaviours - Complete.ens**. This scene is an example of how your scene should look if you have successfully followed all the steps in this tutorial.

Tutorial 3

Posing the character

In this tutorial, you will learn how to use the **Pose Move** and **Pose Rotate** tools to pose characters using dynamic posing. In particular, you will use the pose tools to sit the character into a chair.



Step 1: Loading the scene

In this step we will open an existing scene that contains some elements of our desired final scene.

To load the initial scene:

1. Launch *endorphin*.
2. Select **File > Open Scene…** to browse for a scene. The keyboard shortcut is **Ctrl+O**.
3. Open the scene file **Resources\Tutorials\Tutorial 3 – Posing the Character \Tutorial 3 – Posing the Character - Begin.ens**.

   The scene contains a character, along with a chair in the Environment character made up of a set of collision objects. The goal of this tutorial is to pose the character into the chair.

Modelling the chair

The chair was created by first importing its shape as an **OBJ** file. The OBJ mesh surface is stored in the scene as a graphical object associated with the Environment character. Next, a set of collision objects were created and moved, rotated and scaled in order to fill out this OBJ mesh.

Step 2: Activating the pose tools

In this step we will activate the Pose Move and Pose Rotate tools. These tools are **dynamic posing** tools. These tools work in a similar manner to the Move and Rotate tools, with some differences.

When you use the pose tools, *endorphin* performs **collision detection** on collision objects in the scene. You are not able to move an object so that collision objects intersect. Collision detection is useful when you want to carefully align objects without them intersecting.

Another benefit of the pose tools is that they let you pose characters in a manner similar to **IK (inverse kinematics)** posing. For example, you can pose move or rotate the hand of a character and also affect its shoulder and elbow at the same time.

To activate the pose tools:

1. Before using the pose tools make sure the Time Slider is set to **Frame 0**.
2. Select **Edit > Pose Move** and **Edit > Pose Rotate** to activate the Pose Move and Pose Rotate tools respectively. The keyboard shortcuts are **A** and **S** respectively. Alternatively, you can click the **Pose Move** or **Pose Rotate** buttons on the Main Toolbar.

   When you are using one of the pose tools, the viewport is displayed with a red border.
3. You can quickly toggle between the Pose Move and Pose Rotate tools by double-clicking their corresponding manipulators in the viewports.
4. We can edit the character's pose by dragging and rotating limbs. We can also lock limbs in space to stop them moving once we are happy with their orientation.

Step 3: Working with the pose tools

In this step we will manipulate the character using the **dynamic pose** tools. Unlike the standard Move and Rotate tools, the pose tools use a form of IK (inverse kinematics) posing. If you pose the hand, for example, you also affect other joints such as the shoulder and elbow.

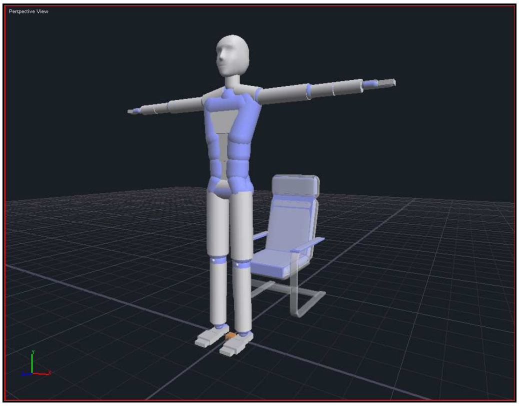
Freezing mass objects is a useful way to limit the extent to which the dynamic pose tools affect other parts of the body. For example, if you freeze the right shoulder mass object, and use the Pose Move tool to move the right hand mass object, the posing will only affect the right arm. In contrast, if no mass objects are frozen, then moving the right hand would potentially affect the entire character.

To pose the character using the pose tools:

1. Activate the Pose Move or Pose Rotate tools.
2. Select the mass object you want to pose, such as the upper leg. The mass object will be displayed in a **red highlight colour** to indicate that it is the posed object. You can pose multiple mass objects at the same time.

   The remainder of the character will be displayed in a **light orange highlight colour**, indicating that these mass objects will be affected by any pose operations that you perform on the selected mass object.
3. Experiment with the Pose Move and Pose Rotate tools. Remember that you can **double-click** on the pose manipulator to toggle between the Pose Move and Pose Rotate tools.
4. If you want to reset the character pose, select **Character > Reset Pose**.

   Alternatively, click the **Reset Pose** button on the Main Toolbar. The Reset Pose command will place the character into its default pose. For the standard *endorphin* simulation character, the default pose is a **T-pose**.



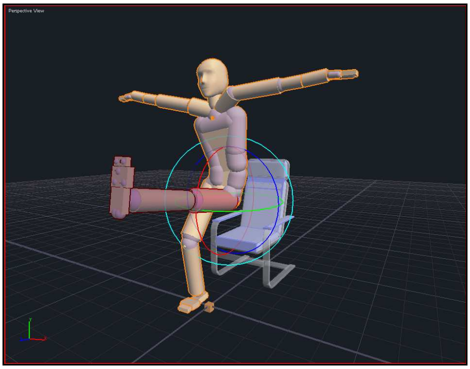
To pose using local coordinates:

- By default, the Pose Move and Pose Rotate tools use the **global** coordinate axis system. However, you can also use these tools in the **local** coordinate system of the mass object that you are posing. When you pose using the local coordinate system, the pose tools work in exactly the same manner, except that they use different X, Y and Z directions.
- To toggle between global and local coordinates, select **Edit > Toggle Coordinate System**. The keyboard shortcut is **X**. This setting affects both the Pose Move and Pose Rotate tools. This setting is **not** changed if you select a different mass object to pose.

To lock mass objects during dynamic posing:

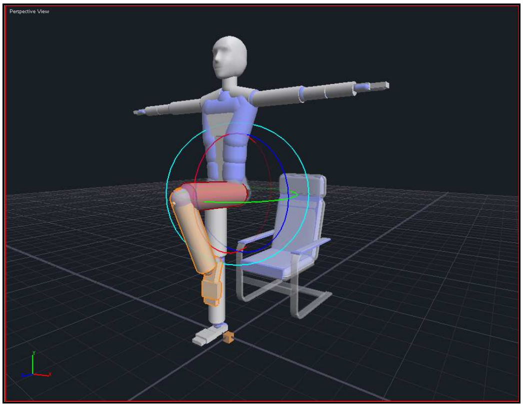
- You can dynamically pose more than one mass object at the same time. When you do this, the relative positions of the selected mass objects are maintained with respect to each other. In effect, you are **locking** the mass objects together for the duration of the pose.
- Typically, you will lock mass objects that are further **down** the joint hierarchy from the posed mass object. Then, when you pose the first mass object, the other mass objects automatically move as well, in an FK (forward kinematics) manner.
- In this scene, hold down the **Ctrl** key and select multiple mass objects, such as the **LeftUpperLegMass** and **LeftLowerLegMass** mass objects. The **first** mass object you select is displayed using the red highlight colour, indicating that it is the **posed** mass object. Any subsequent mass objects that you select are displayed in a dark red highlight colour. This indicates that these mass objects are **locked** to the posed mass object.

- Experiment with using the Pose Move and Pose Rotate tools in conjunction with locked mass objects. Then click **Reset Pose** on the Main Toolbar to reset the character back to its default T-pose.



To freeze mass objects during dynamic posing:

- You can **freeze** mass objects by right-clicking on them in the viewport. Frozen mass objects are displayed using a grey highlight colour. When a mass object is frozen, its position and orientation are locked and are no longer affected by the Pose Move or Pose Rotate tools. If you right-click a frozen mass object, it becomes active again.
- Typically, you will freeze a mass object that is further **up** the joint hierarchy from the posed mass object. Then, when you pose the mass object, the frozen mass object defines the limit of mass objects that are affected by the pose tools. That is, only joints **downstream** of the frozen mass object are affected.
- In this scene, **right-click** on a mass object such as the **LeftUpperLegMass** mass object. This freezes the mass object. Keep in mind that you can right-click on a collision object in order to freeze the corresponding parent mass object. If you then select a mass object **downstream** of the frozen mass object, such as the LeftLowerLegMass mass object, you can pose the left leg in an IK manner without affecting the rest of the character beyond the left leg.
- Experiment with locking the **PelvisMass** mass object and using the Pose Move and Pose Rotate tools to pose the legs and feet of the character. Then click **Reset Pose** on the Main Toolbar to reset the character back to its default T-pose, and ensure that all the mass objects are active.



Step 4: Sitting the character in the chair

In this step we are going to use dynamic pose tools to actually sit the character into the chair.

Using dynamic posing

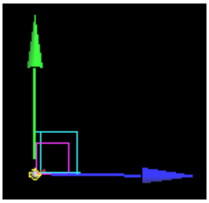
Keep in mind that posing using the dynamic pose tools is different from the standard FK (forward kinematics) and IK (inverse kinematics) posing methods you are probably used to. When you pose a character using dynamic pose tools, the character reacts as though you were actually posing a person physically. This is because the dynamic pose tools actually use a **dynamic simulation** behind the scenes in order to carry out the pose.

Dynamic posing is particularly useful when posing characters in **close proximity** to environment objects. For example, dynamic posing allows you to easily position a character onto a prop object such a cylinder by simply lowering the character onto it. Collisions between the collision objects of the character and the prop will automatically lead to the character's legs moving apart to accommodate the cylindrical shape.

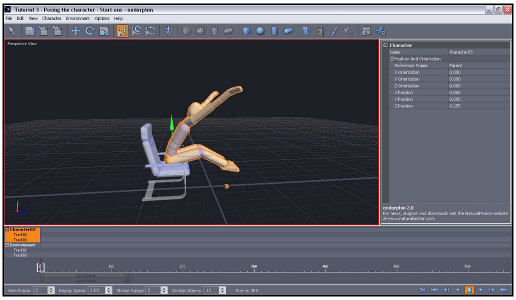
It may take time to get used to working with the dynamic pose tools. However, once you have learned to use them, they can save a lot of time when aligning characters with objects around them. Keep in mind that by **freezing** body parts you can essentially revert to IK posing, which you might be more familiar with.

To pose the character into a sitting position:

1. Activate the **Pose Move** tool.
2. Select the **PelvisMass** mass object.
3. Move the PelvisMass mass object using a **constrained move** in the YZ plane. That is, drag inside the light-blue square between the green Y axis and the blue Z axis. This ensures that as you move the pelvis, it remains in a plane that passes through the centre of the chair.



4. Move the pelvis backwards and downwards until it makes contact with the chair back and seat.

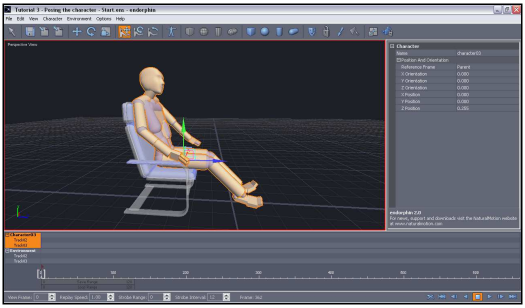

Tutorial 3 - Posing the character - Start.ens - endorphin
File Edit View Character Environment Options Help
Perspective View
Character03
Track02
Track03
Environment
Track02
Track03
Save Range
Loop Range

Character	
Name	character03
Position And Orientation	
Reference Frame	Parent
X Orientation	0.000
Y Orientation	0.000
Z Orientation	0.000
X Position	0.000
Y Position	0.000
Z Position	0.255

endorphin 2.0
For news, support and downloads visit the NaturalMotion website at www.naturalmotion.com

View Frame: 0 Replay Speed: 1.00 Strobe Range: 0 Strobe Interval: 12 Frame: 359

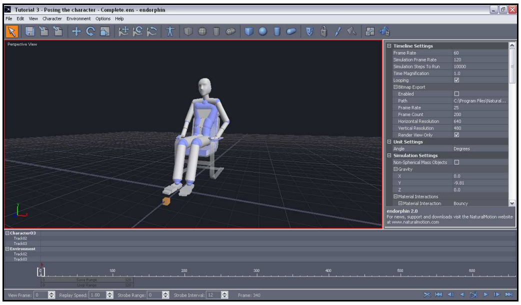
5. Once the pelvis is in place, select the UpperSpineMass mass object, and **right-click** on the PelvisMass mass object to freeze it. The PelvisMass mass object should be displayed in the grey highlight colour, indicating that it is frozen.
6. Use the Pose Move tool to move the upper torso back against the chair back. If any of the lower back parts rotate out of alignment, select them and rotate them back using the Pose Rotate tool. You may need to **lock** or **freeze** other mass objects to do so. Remember that you can toggle the Pose Move and Pose Rotate tools so that they use local or global coordinates by selecting **Edit > Toggle Coordinate System**, or by pressing the **X** key.


Tutorial 3 - Posing the character - Start.ens - endorphin
File Edit View Character Environment Options Help
Perspective View

⊟ Character	
Name	character03
⊟ Position And Orientation	
Reference Frame	Parent
X Orientation	0.000
Y Orientation	0.000
Z Orientation	0.000
X Position	0.000
Y Position	0.000
Z Position	0.255

endorphin 2.0
For news, support and downloads visit the NaturalMotion website at www.naturalmotion.com

7. In a similar manner, use the dynamic pose tools to raise the character's head and lower its feet towards the ground. Also, pose the arms so that the character's hands are placed on the arm rests.


Tutorial 3 - Posing the character - Complete.ens - endorphin
File Edit View Character Environment Options Help
Perspective View

⊟ Timeline Settings	
Frame Rate	60
Simulation Frame Rate	120
Simulation Steps To Run	10000
Time Magnification	1.0
Looping	☑
⊟ Bitmap Export	
Enabled	☐
Path	C:\Program Files\Natural...
Frame Rate	25
Frame Count	200
Horizontal Resolution	640
Vertical Resolution	480
Render View Only	☑
⊟ Unit Settings	
Angle	Degrees
⊟ Simulation Settings	
Non-Spherical Mass Objects	☐
⊟ Gravity	
X	0.0
Y	-9.81
Z	0.0
⊟ Material Interactions	
⊟ Material Interaction	Bouncy

endorphin 2.0
For news, support and downloads visit the NaturalMotion website at www.naturalmotion.com

8. You can save and re-used poses by selecting **Character > Save Pose** and **Character > Load Pose**. You can also use these commands to load poses into **active poses**. This is covered in a later tutorial.

Conclusion

In this tutorial, you have learned how to use the **Pose Move** and **Pose Rotate** tools to pose characters using dynamic posing.

If you have any problems with this tutorial, open the corresponding scene **Resources\Tutorials\Tutorial 3 – Posing The Character\Tutorial 3 – Posing The Character - Complete.ens**. This scene is an example of how your scene should look if you have successfully followed all the steps in this tutorial.

Tutorial 4

Creating environments

In this tutorial, you will start building up the virtual world **environment**.

The default environment is a simple flat ground plane. However, it is easy to build new environment objects such as buildings, walls and furniture, to provide a richer environment for the *endorphin* characters to interact with.

You will also import a mesh to act as a visual guide for building the environment.

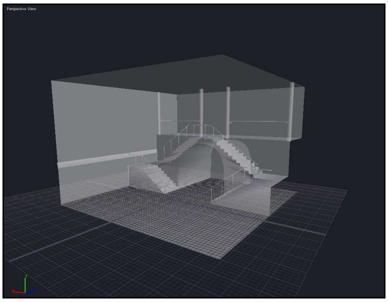
Step 1: Visualising the environment

In this step will we be creating a new scene and importing an OBJ file that contains a polygon mesh that represents the environment objects we will be modelling in later steps.

To import a visualisation mesh:

1. Launch *endorphin*.
2. Select **File > New Scene** to create a new scene, and delete the default character.
3. Select **File > Import**, and browse to the file **Resources\Tutorials\Tutorial 4 – Creating Environments\HallOnly.obj**.
4. Click **OK**, then click **Import** in the Import Options dialog. The OBJ file is imported and a corresponding graphical object is created.

   This graphical object lets you visualise the scene that you want to create. The next step is to fill out this mesh with collision objects so that the *endorphin* characters can collide with the environment.



Step 2: Building the staircase

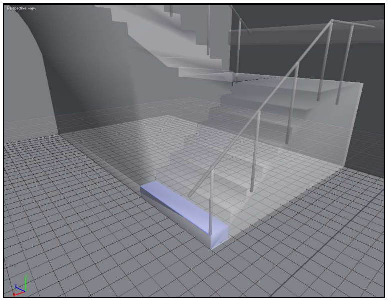
In this step we will be building up the staircase using box collision objects.

To create the first step:

1. Click the **Create Box Collision Object** button on the Main Toolbar. This creates a new collision object at the origin of the environment. The new collision object is initially selected.
2. Use the **Move**, **Rotate** and **Scale** tools to place this box into its correct position at the foot of the staircase. You can also use the Property Editor to set exact values for the position and size of each collision object.

   A good technique when you are positioning collision objects is to watch for intersections between the collision objects and the visualisation graphical object in the viewport. Usually, your goal will be to position collision objects as close as possible to the visualisation surface without penetrating it.

   Note You can double-click on Move, Rotate and Scale tool manipulators to cycle through these manipulators. Alternatively, you can use the **Q**, **W** and **E** keyboard shortcuts to rapidly change the active editing tool.



To create the next step:

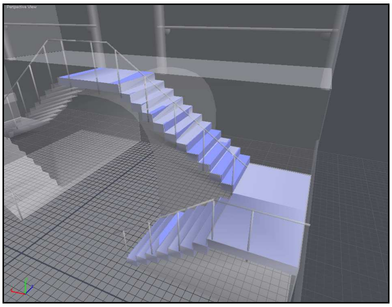
1. Select the first step collision object.
2. Copy-and-paste this object to create a new collision object. You can select **Edit > Copy** and **Edit > Paste** to do this. Alternatively, the keyboard shortcuts are **Ctrl+C** and **Ctrl+V** respectively.

   Note If you find that the visualisation graphical object is making it awkward to select collision objects in the environment, you can **freeze** it so that it is no longer selectable in the viewport. Keep in mind that you can still select frozen objects using the Node View. Graphical objects created from OBJ files will have the same name as the corresponding OBJ file name.

   To **freeze** a graphical object, right-click on the object in the Node View and select **Freeze Selection**. To **unfreeze** a frozen graphical object, right-click on the object in the Node View and select **Unfreeze Selection**. Only graphical objects can be frozen or unfrozen.
3. Use the Move and Rotate tools to position the new step in its correct position as the second step. You should not need to use the Scale tool as the new collision object will have the same shape and size as the original step.

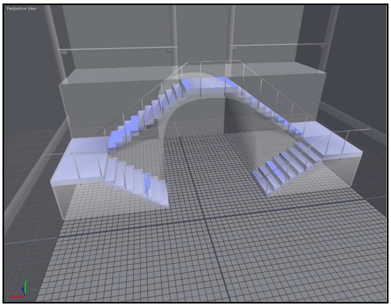
To create one side of the staircase:

1. You can now select **both** of the steps and copy-and-paste them to create two new collision objects.
2. Move these two new steps into position.
3. You can now continue this process by selecting all four steps and copying-and-pasting them. In a similar manner, create all the required steps and place them all into position using the Move and Rotate tools.



To create the entire staircase:

1. The entire staircase is made up of two identical halves. You have already built one half of the staircase. Select all the collision objects you have created, and copy-and-paste this entire set.
2. With this new set of collision objects still selected, use the Rotate tool to rotate them into position, using the visualisation graphical object as a guide.

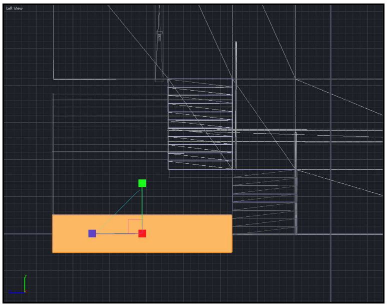


Step 3: Building the archway

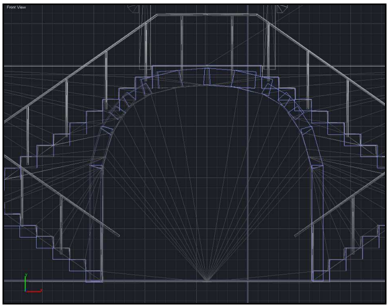
In this step we will be building up an archway under the staircase using box collision objects.

To build the archway:

1. Click the **Create Box Collision Object** button on the Main Toolbar. This creates a new collision object at the origin.
2. Display the Left View by right-clicking and selecting **Left View**. The keyboard shortcut is **L**.
3. Display the viewport in wireframe mode by right-clicking and selecting **Wireframe View**.
4. Use the **Scale** tool to resize the new collision object until its depth matches the depth of the archway.



5. Display the Front View by right-clicking and selecting **Front View**. The keyboard shortcut is **F**.
6. Use the Move, Rotate and Scale tools to position and size the collision object so that it matches one section of the arch.
7. Use the copy-and-paste tools to create an entire set of collision objects to fill out the archway.



Note You could also create this archway by building half the archway, and then using the copy-and-paste and Rotate tools to position a copy of the arch into the correct position.

Step 4: Building a collapsible railing

In this step we will be building a railing that follows the path of the staircase. This railing will be made up of cylindrical collision objects. So far, the staircase and archway that you have built have been made up of collections of **static** collision objects. In contrast, you will be building the railing as a **dynamic** environment object.

Static and dynamic environment objects

By default, when you add collision objects to the environment, they are created as child objects of the **GroundMass** mass object of the Environment character. This mass object does not move during simulations, which effectively locks its child collision objects into place. This approach is useful for creating **static** environment objects such as buildings, which do not need to react to collisions.

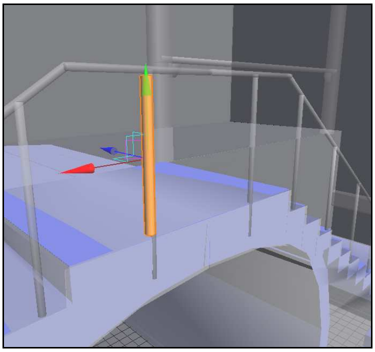
You can also create **dynamic** environment objects. The first step is to create a new mass objects to the environment character. The second step is to add child collision objects to these new mass objects. These collections of collision objects can be involved in collisions, and can develop their own velocity and momentum. This approach is useful for creating objects such as pieces of furniture that are designed to react to collisions during a simulation.

A further technique is to combine sets of dynamic environment objects using constraints. Constraints that combine dynamic collision objects are often called **glue constraints**. By changing the strength and duration of these glue constraints during a simulation, you can generate complex dynamic effects.

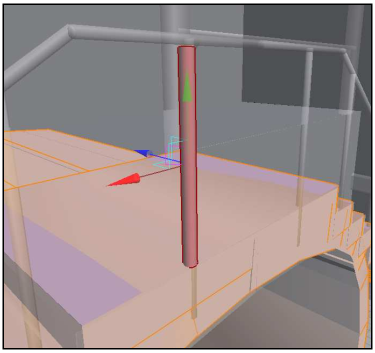
To create a dynamic railing:

1. Display the viewport in shaded mode by right-clicking and selecting **Shaded View**.
2. Click the **Create Cylindrical Mass Object** button on the Main Toolbar. This creates a new mass object at the origin. Note that we are creating a mass object, rather than a collision object.

   Keep in mind that you can always add and remove other collision objects to new mass objects in the Environment character. You just cannot select or remove the implicit collision object.
3. Use the Move, Rotate and Scale tools to position and size this mass object so that it roughly corresponds to the position and size of one of the vertical struts of the railing.



4. Use the **Pose Move** tool and move the mass object so that rests exactly on the top of the staircase. The Pose Move and Pose Rotate tools are particularly useful when you want to align objects, because these tools consider the effect of the collision objects on each other. In contrast, the usual editing tools such as Move, Rotate and Scale ignore the effect of collision objects.



5. Copy-and-paste the new mass object, and use the Move tool to position it so that it is aligned with its neighbouring vertical strut.
6. Copy-and-paste the new mass object twice more, and use the Move, Rotate and Scale tools to position these new mass objects as crossbars sitting on top of each of the vertical struts. You can also use the **Pose Move** tool to accurately sit the crossbars onto the tops of each of the vertical struts, forming a pair of T-shapes.
7. At this stage we have only begun the process of building the entire railing. To **complete** the railing you would repeat the previous steps, cutting-and-pasting existing mass objects and using the Move, Rotate and Pose Move tools to accurately position these mass objects to reflect the shape of the railing.

Step 5: Locking the railing together with a constraint

In this step we will use a constraint to hold together the mass objects that make up the railing. By releasing this constraint at the moment of impact between a simulated character and the railing, we can create the effect that the collision is causing damage to the railing, causing it to collapse.

To lock the railing objects together:

1. With the railing still partially built, **simulate** the scene. You should find that the four mass objects that you have added topple and fall under the influence of gravity.

   You can lock the mass objects together using a **constraint event**. Constraint events are useful when you want to hold together various mass objects until a particular moment in a simulation. This is a useful technique for mimicking the effect of objects breaking as a result of collisions with other objects.

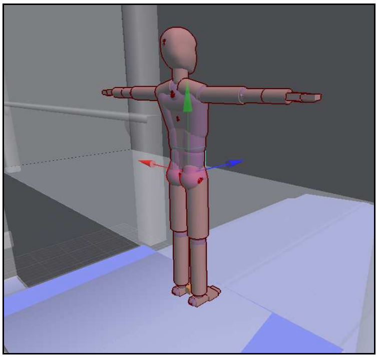
2. Click on the **Clear Frames** button in the Simulation Toolbar to clear the frame buffer.
3. In the **Environment** character timeline in the Timeline Editor, right-click and select **Create Constraint Event**. This creates a new constraint event for the environment. You are automatically placed into Selection mode, which lets you select the mass objects to be constrained.
4. In the viewport, **box select** the mass objects you have created to form the railing. Right-click to exit Selection mode, or press the **Esc** key.
5. Ensure that the constraint event starts at Frame 0. Extend the constraint event so that its end frame is approximately at Frame 500.
6. Simulate again. You should find that this time the mass objects making up the railing will stay fixed in place. When the end frame of the constraint is reached during the simulation, the railing should then collapse under the weight of gravity.

Step 6: Adding a character to the environment

In this step we will add a simulation character to this environment, and create a simulation in which the character collides with the railing, causing it to collapse.

To add a character to the scene:

1. Select **Character > Add Character** to display the Add Character dialog. Choose the Standard Simulation Character and click **OK**.
2. Select the new character and use the Move tool to position the character at the top of the staircase.
3. Activate the **Pose Move** tool. Select the character as a whole using the Timeline Editor, or by selecting the character cube in the viewport. Pose Move the character so that it is positioned exactly on the top step.



To push the character through the railing:

1. Add a **force event** to the character timeline. Ensure that the force is applied at Frame 0. Leave the other force properties at their default settings.
2. Add a **behaviour event** to the character timeline. Set the behaviour name to **Stagger**, and ensure that it also begins at Frame 0.
3. Simulate the scene. The character should take a few steps backwards as it is hit by the force event. After a few steps, the character should then collide with the railing.
4. Stop the simulation after the character collides with the railing, and move the Time Slider back to the frame at which the character first makes contact with the railing.

5. elect the constraint event, and adjust its end frame to the current frame.
6. Simulate again. The character should now stagger backwards, collide with the railing, and then fall through the collapsing railing and down to the ground. By timing the end of the constraint with the moment of collision, the railing appears to collapse as a result of the collision with the character.
7. Experiment with the addition of other behaviour events to the character timeline. For example, you may want to add a **Writhe** behaviour while the character is falling, and then a **Catch Fall** behaviour as the character collides with the ground.

Conclusion

In this tutorial, you have created a complex environment using mass objects and collision objects to create static and dynamic scene objects. In addition, you have used constraint events to combine multiple environment mass objects into single compound objects to create the effect of collapsing scene objects.

If you have any problems with this tutorial, open the corresponding scene **Resources\Tutorials\Tutorial 4 – Building Environments\Tutorial 4 – Building Environments - Complete.ens**. This scene is an example of how your scene should look if you have successfully followed all the steps in this tutorial.

Tutorial 5

Creating characters

In the first four tutorials, you have created scenes using the **standard** *endorphin* simulation character. This character represents a typical adult male human.

However, you are not limited to creating scenes with this character only. You can use **Character Edit Mode** to **reshape** the standard character to create your own custom characters.

In this tutorial, you will use Character Edit Mode to create a custom character that matches the shape and size of an external character stored in an animation data file. You will then use **dynamic motion transfer** to map the motion data stored in this external animation file onto your reshaped *endorphin* character.

Before you begin

Before we begin this tutorial, it is important to understand some key concepts associated with the *endorphin* animation pipeline.

Reference Character

A reference character is an endorphin character that has been generated by **importing** a character stored in an external animation file. Reference characters can be generated from FBX, BVH, XSI, ASF or Vicon V files that you have exported from Maya, 3dsMax or other animation system. Reference characters have the **same** joint hierarchy and naming conventions found in the original file.

You do not actually use reference characters in *endorphin* scenes. Rather, you will typically **reshape** a simulation character to match its corresponding reference character, and use the reshaped character in the scene.

Reference character files have the extension **_ref.nmc**.

Simulation Character

A simulation character is any *endorphin* character that has been derived from the standard *endorphin* simulation character. That is, it has the same joint hierarchy and object naming conventions as the standard simulation character.

However, simulation characters may have been **reshaped** to match a **reference** character. A reshaped character may have different joint lengths, orientations and joint limits to the standard simulation character. Also, a reshaped character may have different mass object or collision object shapes, positions or orientations.

Simulation character files have the extension **_.nmc**.

Simulation-Reference Character Pair

When you are using *endorphin* as part of an animation pipeline, you will typically create **pairs** of associated *endorphin* characters. Each pair will be made up of a **reference** character that you have created by exporting the character from your original animation system, and a corresponding simulation character that you have derived from the standard *endorphin* character, and reshaped to match its reference character.

Dynamic Motion Transfer

The process of mapping motion from the reference character to the simulation character, or from the simulation character to the reference character, using physical dynamics.

When you are **importing** animation data, you will import it onto a reference character and dynamically transfer it to the corresponding simulation character in the scene.

When you are **exporting** animated data that you have generated in *endorphin*, you will dynamically transfer it from the simulated character to its corresponding reference character, and then export the animation file based on the motion of the reference character.

Step 1: Creating new characters

In this step, we will create a simulation-reference character pair in *endorphin* using Character Edit Mode. The reference character will be based on the skeleton contained in an FBX file.

To create a new simulation-reference character pair:

1. Launch *endorphin*.
2. Select **File > New Character…**. This displays the **New Character** dialog.

3. In the **Create Character Using** group, ensure that **Standard Simulation Character** is selected. This ensures that the new simulation character will be based on the standard simulation character. You must derive new simulation characters from this character, **or** from other characters that have themselves been derived from this character.
4. Turn on the **Create Reference Character Using** setting, and click the Browse button to select the file **Resources\Tutorials\Tutorial 5 – Creating Characters (Basic)\AM_T-pose.fbx**. This ensures that the new reference character will be created from the skeleton found in this file. It is strongly recommended that you only create reference characters from animation files in which the character is in its **T-pose**. This will make the animation pipeline more reliable.
5. Click **OK**. The **Import Options** dialog appears, which lets you specify settings for use when creating the reference character. Keep all the settings at their default values. In the node hierarchy tree, select the **Hips** node. This ensures that the new reference character will be created with its root joint based on the Hips joint in the source animation file. It is important that you select the correct node to use as the basis for your reference character.
6. Click **OK**. A new simulation-reference character pair will be generated, and you will automatically be placed in Character Edit Mode. In this mode, there is no Timeline Editor or Simulation Toolbar. Also, some of the commands in the Main Toolbar and Menu Bar will be different. For example, you cannot create mass objects or timeline events in Character Edit Mode.

Step 2: Adjusting the root position

In this step, we will begin the process of **reshaping** the simulation character to match its corresponding reference character. This process involves modifying the joint lengths and orientations of the simulation character so that they match its reference character. The first step is to ensure that the **root joint** position of the simulation character exactly coincides with the root joint position of its reference.

Using the Move tool to snap joints

Snapping is a very useful way for quickly positioning joints in the simulation character to match joint positions of the reference character. You can snap any joint except the root joint. You can also snap reference character joints to simulation character joints, although this is rarely done. Usually, you will be snapping the simulation character to the reference character.

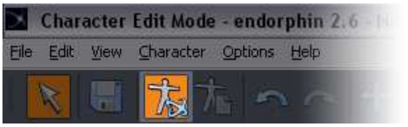
Snapping only works when you are moving a joint using the Move tool in its **unconstrained** mode. If you are constrained dragging along any axis or in any plane, snapping will not occur. The snap threshold depends on the apparent relative positions of the source and target joints as they **appear on the screen**. This means you do not need to worry about how close the joints are in 3D space. As long as they appear to overlap on the screen, they can be snapped to each other.

Keep in mind that snapping does **not** generate a permanent connection between the snapped joints. If you adjust the position of a joint, you may also affect the positions of any the joints downstream of that joint, regardless of whether they have already been snapped to reference character joints.

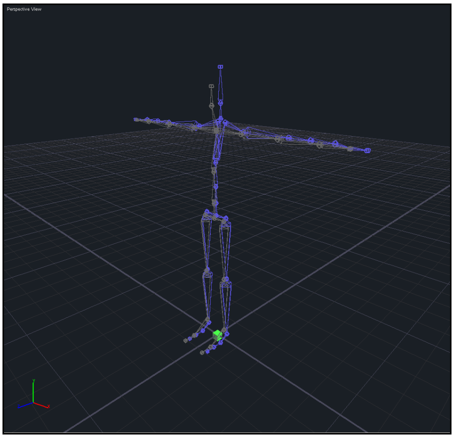
To match the root position of the simulation and reference characters:

1. Right-click in the viewport and select **Skeletal View**. This mode is useful because most of the reshaping work in this tutorial involves changing joint lengths and orientations.
2. You should be able to identify two skeletons in the viewport. The **active** character skeleton will be displayed in blue. The **inactive** character skeleton will be displayed in grey. At any point in time, only one of the characters is active. You cannot select or move the inactive character.

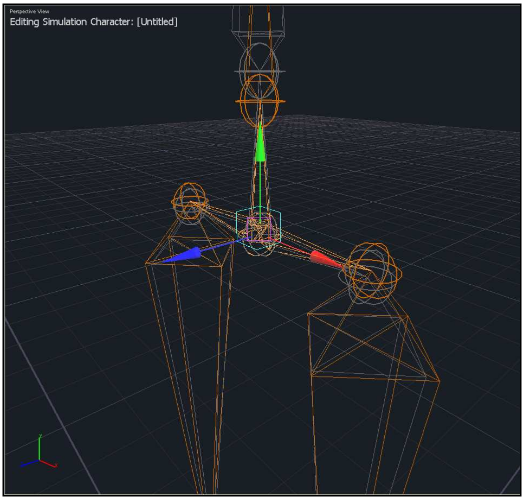
   By default, the simulation character will be active. In Character Edit Mode, there are two special buttons in the Main Toolbar that you use to control which character is active.

3. You are now ready to reshape the simulation character to match its reference character. The viewport should now look like this:



4. Zoom in on the hips of the two characters.
5. Activate the **Move** tool.
6. Select the **root** joint of the simulation character.
7. Move the root joint using the **purple square** manipulator at the centre of the Move tool manipulator. This is a free, unconstrained move. Drag the joint towards is corresponding root joint on the reference character. When you get close to this joint, the simulation character joint should **snap** to the position of the reference character joint.



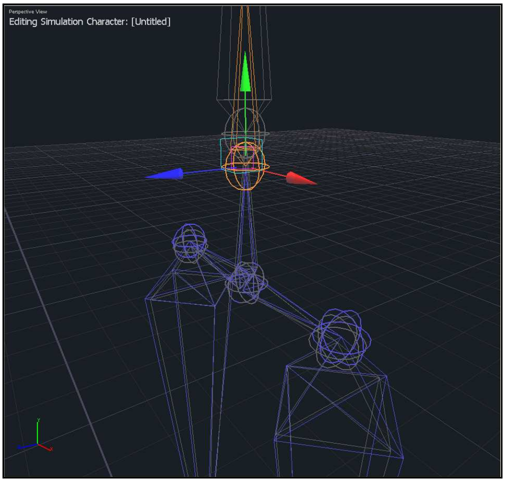
Step 3: Reshaping the joint hierarchy

In this step, we will continue the process of **reshaping** the simulation character to match its corresponding reference. Starting at the simulation character root joint, and moving outwards in the direction of its extremities, we need to adjust the length and orientation of each joint in the simulation character to match the corresponding joint of the reference character.

In some cases, there will be **no direct correlation** between the joints of the two characters. For example, the reference character may have more or less spine joints than the simulation character, or the reference character may not have forearm twist joints. You will need to make decisions about how best to adjust the joints of the simulation character to best match its reference.

To reshape the joints of the simulation character:

1. Select the **LowerSpineJoint** of the simulation character.
2. If the Move tool is not active, activate it. Move this joint using the **purple square** manipulator at the centre of the Move tool manipulator. This is a free, unconstrained move. Drag the joint towards is corresponding **Chest** joint on the reference character. When you get close to this joint, the simulation character joint should **snap** to the position of the reference character joint.

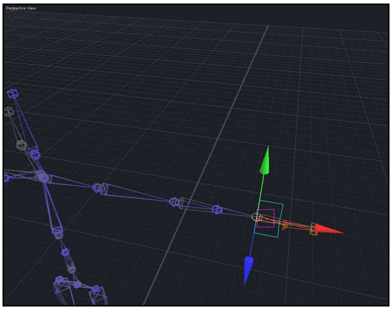


3. Select the **MiddleSpineJoint** of the simulation character, and use the Move tool to position it halfway up the next bone of the reference character. You cannot use joint snapping in this case, because there is no corresponding joint in the reference character to snap to. This is because the reference character only has **two** spine joints, whereas the standard

endorphin simulation character—and any other character derived from it—have **three** spine joints.

When you are using the Move tool and you do **not** want to snap, it is good practice to avoid performing unconstrained moves. This ensures that you do not inadvertently issue a request to snap.

4. Working up the spine, snap all the other joints into place up to the head joint.
5. Select the **LeftClavicleJoint** and snap it into place. In our example, this is to the **same** joint that the **LowerNeckJoint** was snapped to. Work your way down the arm to the **LeftElbowJoint**, snapping joints together.
6. Leave the **LeftForearmTwistJoint** in place. Snap the **LeftWristJoint** to the corresponding wrist joint of the reference character. As with the MiddleSpineJoint, this to account for the fact that the reference character does not have a corresponding forearm twist joint.



7. Next, select the **LeftHipJoint** and snap it into place.
8. Continue working down the left leg until you reach the **LeftAnkleJoint**.

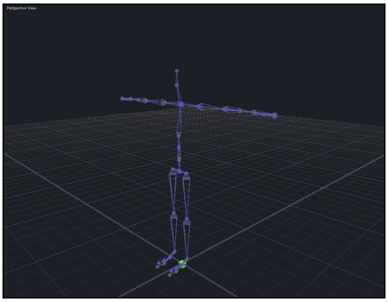
Step 4: Mirroring the reshaped joints

In this step we will copy the changes we made in Step 3 from one side of the character to the other.

Once you have made changes to one side of a simulation character, it is useful to quickly copy them to the other side of the character, since most characters are broadly symmetric. Character Edit Mode supports a set of **mirror tools** to make it easy to copy changes across from one side of a character to another.

To copy changes from one side of the simulation character to the other:

1. So far, we have only edited joints on the left-hand side of the character. To copy these changes to the right-hand side of the character, select **Character > Mirror Left To Right**. Alternatively, click the **Mirror Left To Right** button on the Main Toolbar. The keyboard shortcut is Alt+LeftArrow.
2. After you have mirrored the character, you should find that the simulation and reference character skeletons now match:



Reshaping mass objects and collision objects

So far, we have not explicitly adjusted the position or size of any mass object or collision object. However, if you display the viewport in Shaded View, you will note that the mass objects and collision objects of the simulation character have been **automatically** resized to match the new size of their parent joints. Also, the mirror tools have automatically copied these changes.

After you are have finished reshaping the joint hierarchy of the simulation character, you may want to modify the positions, shapes or sizes of any of its mass objects or collision objects, or add new collision objects to this character. This process is described in **Tutorial 11 – Creating Characters (Advanced)**.

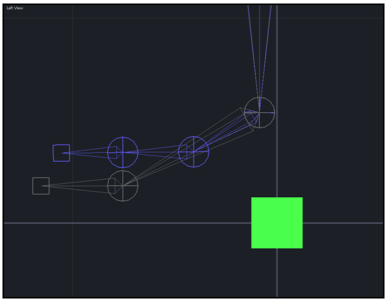
Step 5: Adjusting the foot joints

In this step, we will fine-tune the joints in the feet of the simulation character.

endorphin simulation characters have fairly complex feet that are composed of three joints. This is to ensure that during simulations, characters avoid the **foot-**

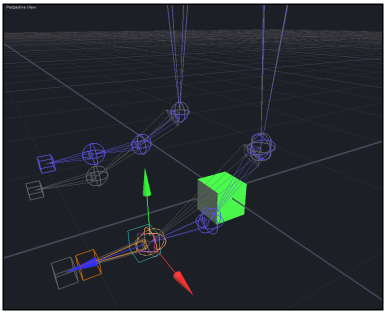
slip problem that is a common problem in animation. Highly articulated feet help ensure good foot-ground contact.

Simulation characters always have three foot joints. These are the **AnkleJoint**, **MidFootJoint** and **ToesJoint**. Reference characters usually have simpler feet. For example, the reference character in this tutorial only has two foot joints. These are the **Ankle** and **Toe** joints.



To adjust the foot joints of the simulation character:

1. Ensure that the simulation character is active.
2. Select the **LeftMidFootJoint**.
3. Use the **Move** tool to move this joint down along the Y-axis in the global coordinate system, so that it is aligned with the toe joint of the reference character.



4. Use the **Move** too to **snap** the toe joint into place. Note that the simulation character **LeftAnkleJoint** has now been rotated. It occurred when you moved the **LeftMidFootJoint** down.
5. Click the **Mirror Left To Right** button on the Main Toolbar to copy these changes to the right foot.

Step 6: Restoring foot-ground contact

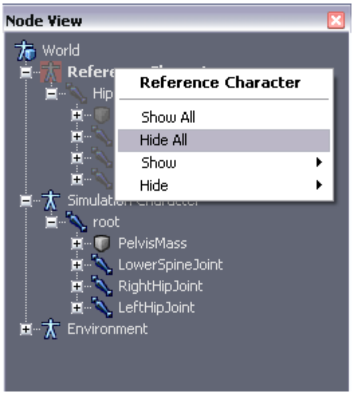
In this step, we will fine-tune the positions of mass objects and collision objects in the feet of the simulation character.

The standard *endorphin* simulation character foot has **seven** collision objects that make contact with the ground. These collision objects play an important role in ensuring that there is no foot-slip during simulations. After reshaping the joints in Character Edit Mode, you may find that the mass objects and collision objects of the foot need to be repositioned to ensure maximum ground-foot contact.

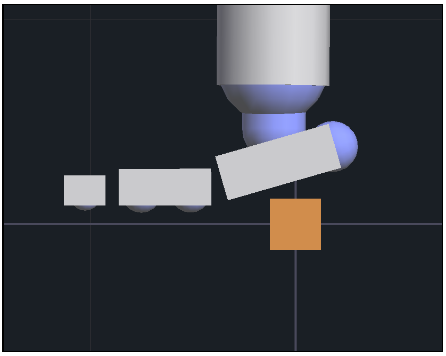
For example, many reference characters have higher **ankle** joint positions than the standard simulation character. After reshaping the simulation character to match its reference character, it may then be necessary to move the mass objects and collision objects in the **heel** back down to contact the ground again.

To adjust the mass objects and collision objects of the simulation character feet:

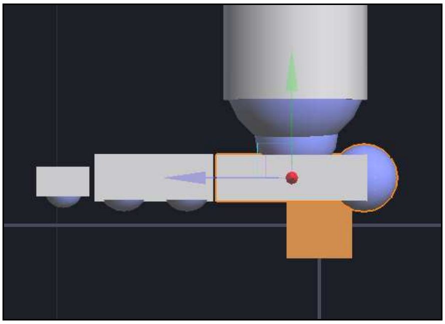
1. Right-click in the viewport and select **Shaded View**.
2. The simulation and reference characters are superimposed, which can make them difficult to work with individually. It is usually easier if you hide the character that you are not working with. To hide the reference character, right-click on the Reference Character node in the Node View and select **Hide All**.



3. Right-click in the viewport and select **Left View**. At the moment, the left foot of the simulation character will not have full contact with the ground. You should find that the mass object and collision objects of the heel are currently raised off the ground.



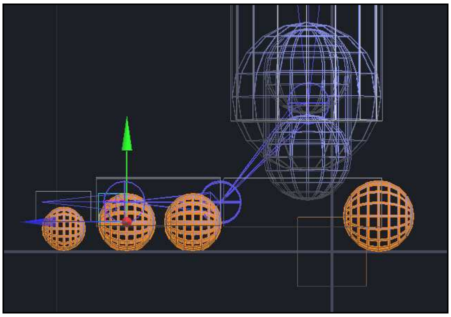
4. Activate the Rotate tool, and rotate the **LeftHeelMass** object down so that the **LeftHeelCollision01** and **LeftHeelCollision02** objects are aligned with the other five collision objects in the foot.



5. The next step is to reposition all seven collision objects making up the left foot so that they make contact with the ground. Select all of these collision objects using box selection, or by holding down the **Ctrl** key and selecting each collision object individually.

6. Activate the **Move** tool and move these collision objects downwards so that they sit exactly on the ground plane.

   Note When modifying the positions of mass and collision objects in the feet, make sure you do not inadvertently modify the positions of any joints. Moving a joint may result in the loss of a rig node connection that you had set up in an earlier step.



7. Click the **Mirror Left To Right** button in the Main Toolbar to copy these changes to the right foot.

Reshaping simulation characters to match arbitrary reference characters

The simulation character is now fully reshaped to match its reference character. You can use similar steps to reshape the standard simulation character to match most humanoid-shaped reference characters.

Keep in mind that *endorphin* behaviours work most reliably when the simulation character is of roughly **human dimensions**. You can use *endorphin* with a range of body sizes—from 3-foot midgets to 12-foot giants—although you may need to fine-tune the character mass distributions to get the more extreme body sizes working correctly.

In some cases, you can mimic extremely large characters by replacing them with smaller characters, using reduced gravity and inertial damping to slow down joint motion.

Step 7: Rigging the reference character

In this step we will view the **connections** between the reference character to the simulation character. The process of creating these connections is sometimes called **rigging** the reference character.

Introducing rig nodes

Rigging is the process of connecting mass objects of each character to corresponding **rig nodes**. When one character **drives** another character using dynamic motion transfer, the actual process involves two steps. Firstly, the motion of the **source** character mass objects drives their corresponding rig nodes; secondly, the motion of these rig nodes drives the motion of their corresponding mass objects on the target character. You can generate different effects by specifying the strength with which mass objects are connected to their rig nodes.

Usually, every rig node is connected to a mass object, but not every mass object is connected to a rig node. This is because most characters have a lot more mass objects than there are rig nodes. Rig nodes are placed at **key locations**, such as the pelvis, shoulders, hands and feet. The motion of intermediate, unconnected mass objects on the target character is generated based on the motion of the connected mass objects and the effects of joint limits.

The mass objects of the standard simulation character are **already** connected to rig nodes. This means that any characters you derive from the standard simulation character are also connected. However, when you create new reference characters, you have to manually add these connections.

Automatic rig node connections

In previous versions of *endorphin*, rigging the reference character was a **manual** process.

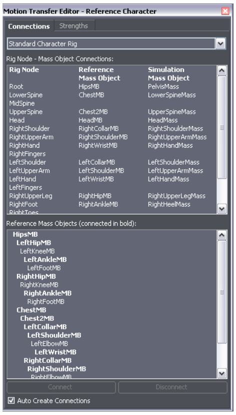
With *endorphin* 2.6, this process has now been automated. Each time you **snap** a simulation character joint to its corresponding reference character joint, *endorphin* now automatically identifies the child mass objects of the snapped joints, and ensures that the reference character mass object is connected to the same rig node as the simulation character mass object.

However, you can always turn **off** the automatic connection mode if you want to make any manual changes to the connections.

To view mass object connections:

1. If the **Motion Transfer Editor** is not visible, select **View > Motion Transfer Editor** to display the **Motion Transfer Editor**. The keyboard shortcut is **M**. The Motion Transfer Editor is automatically displayed when you first enter Character Edit Mode.

   The Motion Transfer Editor contains a **Connections** page that contains a list of rig nodes, along with corresponding lists of mass objects for the reference character and simulation character.



Motion Transfer Editor - Reference Character

Connections | Strengths

Standard Character Rig

Rig Node - Mass Object Connections:

Rig Node	Reference Mass Object	Simulation Mass Object
Root	HipsMB	PelvisMass
LowerSpine	ChestMB	LowerSpineMass
MidSpine		
UpperSpine	Chest2MB	UpperSpineMass
Head	HeadMB	HeadMass
RightShoulder	RightCollarMB	RightShoulderMass
RightUpperArm	RightShoulderMB	RightUpperArmMass
RightHand	RightWristMB	RightHandMass
RightFingers		
LeftShoulder	LeftCollarMB	LeftShoulderMass
LeftUpperArm	LeftShoulderMB	LeftUpperArmMass
LeftHand	LeftWristMB	LeftHandMass
LeftFingers		
RightUpperLeg	RightHipMB	RightUpperLegMass
RightFoot	RightAnkleMB	RightHeelMass
RightToes		

Reference Mass Objects (connected in bold):

```
HipsMB
  LeftHipMB
    LeftKneeMB
      LeftAnkleMB
        LeftFootMB
  RightHipMB
    RightKneeMB
      RightAnkleMB
        RightFootMB
  ChestMB
    Chest2MB
      LeftCollarMB
        LeftShoulderMB
          LeftElbowMB
            LeftWristMB
      RightCollarMB
        RightShoulderMB
          RightElbowMB
```

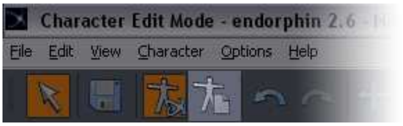
Connect | Disconnect

[✓] Auto Create Connections

2. The list of mass objects in the lower pane reflects the **active** character. Where a mass object name is displayed in **bold**, this indicates that the mass object is connected to a rig node.

   You can change the active character by clicking the **Activate Reference Character** button on the Main Toolbar to activate the reference character, or by clicking the **Activate Simulation Character** button on the Main Toolbar to activate the simulation character.

   You will usually have the reference character active, since the simulation character—and any characters derived from it—are already pre-rigged.



3. If you select a rig node in the **upper** pane, its corresponding mass object (if it has one) is automatically selected in the **lower** pane.

   Similarly, if you select a mass object in the **lower** pane, its corresponding rig node (if it has one) is automatically selected in the **upper** pane.

To edit mass object connections:

1. The **Auto Create Connections** checkbox at the bottom of the Motion Transfer Editor is **on** by default. When this setting is on, mass objects are automatically connected to rig nodes when joints are snapped together in Character Edit Mode. If you want to modify any rig node connections, it can be useful to turn this setting off. This avoids an automatically-created connection from overriding a manual change that you might have made.

2. To disconnect an existing connection, select a rig node or mass object in one of the panes. If the selected entity has a corresponding connection,

the **Disconnect** button is enabled. Click this button to disconnect the selected mass object from the selected rig node.

3. To create a new connection, select a rig node from the **upper** pane, and then select a mass object from the **lower** pane. The selected mass object should not already be connected to another rig node. (If the mass object is already connected to another rig node, first click the **Disconnect** button to disconnect it.) Click the **Connect** button to connect the selected rig node to the selected mass object.
4. If you make changes to the connections on one side of a character, you can use the **Mirror** tools to mirror the connections to the other side of the character. However, keep in mind that the Mirror tools will only modify connections if the **Auto Create Connections** setting is turned on.

Step 8: Saving the new characters

In this step we will save the simulation-reference characters to **.nmc** character files.

To save the new characters:

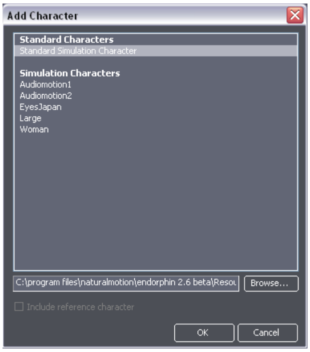
1. Before exiting Character Edit Mode, ensure that the simulation character is visible. To do so, right-click on the **Simulation Character** node in the Node View and select **Show All**. If a character is saved in its hidden state, it will initially be displayed in this hidden state when added to a scene, which can be confusing.
2. Finally, select **File > Save Character As…** to display the **Save As** dialog. You can change the current folder by clicking the Browse button. Specify a character name, keeping in mind that you can add spaces to character names. Do not add a file extension. For example, if you name the character **RiggingTutorial**, *endorphin* will save the simulation character as **RiggingTutorial.nmc**, and will save its reference character as **RiggingTutorial_ref.nmc**. The viewport display label will update to reflect the name of the character.

Step 9: Importing animation

In this step we will add the simulation character to a scene, and then **import** animation onto this character using its reference character as a **dynamic motion transfer** intermediary character.

To add the new simulation character to a scene:

1. Launch *endorphin*.
2. Select **File > New Scene**. Select the default character using the Timeline Editor and delete it.
3. Select **Character > Add Character** to display the **Add Character** dialog. Alternatively, click the **Add Character** button on the Main Toolbar. Select the **RiggingTutorial** character from the list of Custom Characters. You may need to browse to change the active folder. Click **OK**.



C:\program files\naturalmotion\endorphin 2.6 beta\Resou

To import animation using dynamic motion transfer:

1. Select the **RiggingTutorial** character using the Timeline Editor.
2. Select **File > Import**, or click the **Import** button in the Main Toolbar. The keyboard shortcut is **I**. This displays the **Select File To Import** dialog.
3. Browse to the **Resources\Animation\Audiomotion1\FBX** folder, and select any of the FBX animation files in this folder. All of these FBX files contain animation that has been created using the same skeleton that the **RiggingTutorial_ref.nmc** reference character was derived from in Step 2.
4. Click **Open** to display the Import Options dialog.
5. The node hierarchy in the Import Options dialog displays the joint hierarchy of the character in the FBX file. Select the **Hips** node from this tree. This will ensure that only motion of the Hips joint, and any of its child joints, is imported. The motion of any other nodes in the FBX file is ignored.
6. To use dynamic motion transfer, turn on the **Transfer From Reference Character** setting, and then click the Browse button to display the **Open Reference Character** dialog. Choose the **RiggingTutorial** character. This

will select the corresponding **RiggingTutorial_ref.nmc** reference character. Click **OK**.

7. Leave the other Import Options set to their defaults, and click **OK**. Animation from the FBX file will be imported onto the reference character, and then dynamically mapped onto the simulation character. This process occurs automatically. The reference character is temporarily created and destroyed without ever appearing in the scene.

   Keep in mind that the import process can take some time, particularly if there are a lot of frames in the animation file.

8. When the animation import is complete, you will see a new animation event added to the timeline of the simulation character. This animation event contains the imported animation data **applied to the simulation character skeleton**.

Conclusion

In this tutorial, you have learned how to create new simulation-reference character pairs, and how to reshape simulation characters to match their reference characters. You have learned about Character Edit Mode and its various tools, such as the snap tool, and successfully used dynamic motion transfer to import motion from an FBX file onto a simulation character via its reference character.

To learn more about creating and editing characters, see **Tutorial 11 – Character Creation (Advanced)**. To learn more about dynamic motion transfer, see **Tutorial 13 – Advanced Motion Transfer And Prop Interaction**.

Tutorial 6

Creating props

In this tutorial, you are going to create more complex scene objects using collections of mass objects and collision objects saved as three separate **prop characters**.

You are then going to add instances of these prop characters to the scene that you made in **Tutorial 4 – Creating Environments**. You do not need to have worked through Tutorial 4, however, as the final scene from that tutorial has been copied as the initial setup scene for this tutorial.

Before you begin

Before we begin this tutorial, it is important to understand some key concepts associated with *endorphin* environments and prop characters.

Environment objects

In **Tutorial 4**, you created a complicated environment by adding **collision objects** to the Environment character. Adding collision objects directly to the environment is a quick way to create static, rigid structures in your scene.

You also added some **mass objects** to the environment, and attached collision objects to these mass objects. Adding mass objects to the environment is a good way to create dynamic environment entities—such as pieces of furniture—that are mobile and not directly attached to the world. These dynamic objects can gain velocity and momentum as a result of forces and collisions.

Prop characters

In this tutorial, you will use Character Edit Mode to create new prop characters.

Prop characters are collections of joints, mass objects and collision objects that represent **scene objects**. A prop character might be a simple object such as a piece of furniture, or a complex articulated object, such as a vehicle. A prop character can even represent an animal, such as a horse. Prop characters are stored in endorphin **.nmc** character files.

One advantage of creating prop characters is that you can add them to multiple scenes. Another advantage is that you can import animation onto prop characters, which allows you to create much more complex settings for your simulation characters. For example, you might create a horse prop character, add it to a scene, and then import a gallop animation cycle onto the horse. You could then place a simulation character on the horse and simulate the scene. During the simulation, the galloping motion will transmit forces to the simulation character.

Adding timeline events to prop characters

You can apply any type of event—other than behaviour events—to prop characters. Not only can you apply forces, constraints and sever events to a prop character, you can also apply active poses. You can even use active animation events and motion transfer events to reuse animation data in a dynamic context.

You cannot use behaviour events on prop characters. This is because the *endorphin* Behaviour Library has been specifically designed for use with bipedal humanoid characters derived from the *endorphin* standard simulation character.

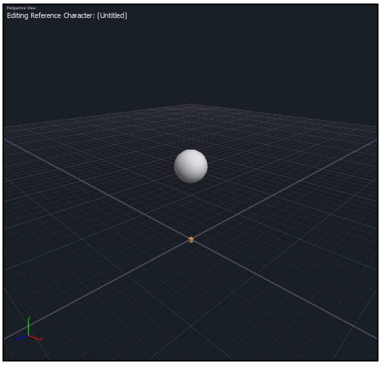
Step 1: Building a chair

In this step we will build a new prop character that represents a **chair**. It will be composed of a single joint and a single mass object, with several collision objects used to fill out its shape.

To create a new prop character:

1. Launch *endorphin*, and select **File > New Character…**.
2. Select the **Standard Prop Character** from the Standard Characters list, and click **OK**.

   The Standard Prop Character contains a single joint and single mass object, and is useful for solid prop objects. You cannot create additional joints or mass objects in Character Edit Mode. To create more complex, articulated characters, you need to create them by importing a skeleton contained in an FBX, BVH, XSI, ASF or Vicon VSK file. You will create a prop character using this approach in the next step.



3. You have now created a new **prop character**. This character is based on the Standard Prop Character and appears as a single spherical mass object. There is no corresponding simulation character, and so the reference character is permanently active when you are editing prop characters. Note that prop characters are considered to be a type of reference character, as they are created to represent equivalent objects in your source animation scenes. Prop characters are saved with a **_ref.nmc** file extension.

4. Display the **Node View** to examine the structure of this character. You will note that the prop character has a single **root** joint, a single **rootMass** mass object and a single **RootCollision** collision object. You cannot delete this joint or mass object. However, you can add and remove collision objects.

To add a graphical object representing the prop shape:

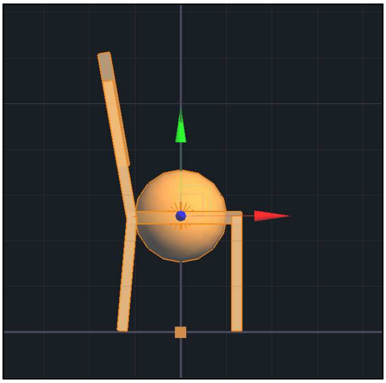
1. Delete the **RootCollision** collision object.

2. Select the **rootMass** mass object.

3. Select **File > Import** and browse to select **Resources\Tutorials\Tutorial 6 – Creating Props\HallChair.obj.** Do not modify any of the default import options. Click **OK**. This imports the OBJ mesh and creates a corresponding graphical object as a child of the selected mass object. This graphical object represents the outer surface of the chair that you are going to fill out using collision objects.

   Before creating any new collision objects, the first step is to ensure that the mass object is placed at the centre of mass of the chair. When you are building props, you do not need a mass object for each collision object. The equivalent dynamic properties of the compound object can be approximated by a single mass object, provided that it is located at the centre of mass of the entire object, and has a mass equivalent to the mass of the entire object.

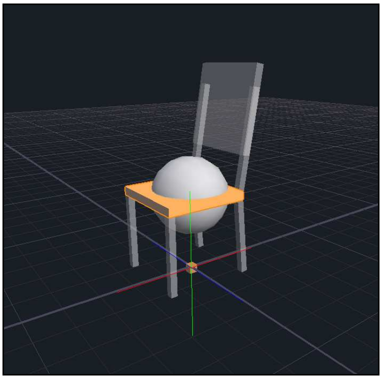
4. Activate the **Move** tool.
5. Select the graphical object, and move it until the mass object is located roughly at the centre of mass of the chair.
6. Select the mass object, and move it vertically so that the chair legs of the graphical mesh make contact with the floor. Keep in mind that when you move the mass object, the graphical object also moves. This is because the graphical object was created as a **child** of the mass object.



To fill out the prop with collision objects:

You can now start adding collision objects to fill out the chair shape represented by the graphical object. Before you can create a collision object, you must first select the mass object that will become the parent of the new collision object.

1. Select the **rootMass** mass object.
2. Click the **Create Box Collision Object** button on the Main Toolbar. This creates a new box collision object.
3. Use the **Move**, **Rotate** and **Scale** tools to adjust the size and position of this collision object until it coincides with the chair seat.



4. Add more collision objects using the same approach. Remember that each time you create a new collision object, the parent mass object must be preselected. After you have created each new collision object, use the Move, Rotate and Scale tools to adjust the size and position the object until it coincides with its corresponding feature of the graphical object.

5. When you have finished modeling the chair, select **File > Save Character**. In the **Save Character** dialog, name the prop character **Chair** and click **OK**.

How many collision objects should I add?

Choosing how many collision objects to add depends on the type of prop and its intended use. You can usually ignore **small** features if they are unlikely to affect the way the prop interacts dynamically with other objects. Each additional collision object will slightly affect the simulation rate, so it is good practice to attempt to fill out props using the minimum number of collision objects.

If you are modeling a shape with a **complex** surface, and there is going to be a lot of contact with this surface during the simulation, then you may need to create many more collision objects to accurately reflect this shape; otherwise your resulting animation may contain object penetration.

These guidelines also apply when you are creating **simulation** characters derived from the standard simulation character.

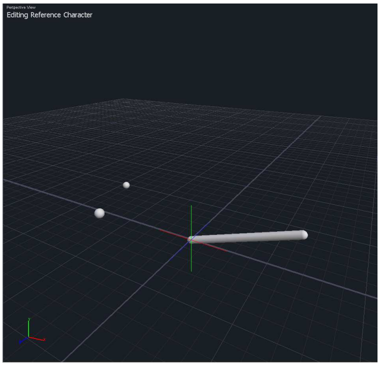
Step 2: Creating a collapsible table

In this step we will build a new prop character that represents a **table**. Unlike the chair, the table will be made up of multiple joints and mass objects. This approach allows you to create the effect of breaking objects during a simulation. This technique was first introduced in **Tutorial 4 – Creating Environments**.

To create a table prop character using an imported character:

1. Select **File > New Character…**.
2. In the New Character dialog, turn **off** the Create Character Using setting, and turn **on** the Create Reference Character Using setting.

   You are going to create a new prop character using a character imported from an FBX animation file, rather than using the Standard Prop Character as the basis of the prop character. This approach is required when you want to create prop characters with more than one joint or mass object. This is because you cannot create new joints or mass objects in Character Edit Mode.
3. Browse to select the **Resources\Tutorials\Tutorial 6 – Creating Props\HallTableBroken.fbx**.
4. In the Import Options dialog, select the **root** node in the node tree hierarchy, and then click **OK**. This ensures that the new prop character will be created with its root joint based on the **root** joint in the source animation file. It is important that you select the correct node to use as the basis for your prop character.

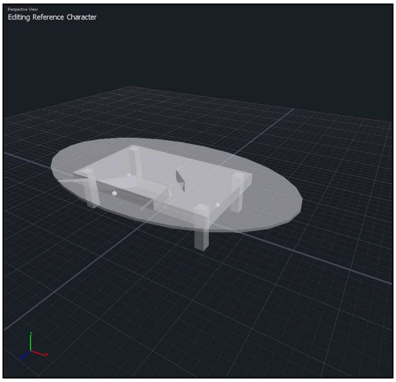
5. You have now created a new prop character based on the character imported from this FBX file. Display the **Node View** to examine the structure of this character. You will notice that the root joint has three

child joints. These joints will form the basis of three components making up the entire table.

6. Select the **rootMass** mass Object and delete it. This mass object is not required for the table.

To add a graphical object representing the prop shape:

1. Select the reference character. One way to do this is to select the Reference Character node in the Node View. Another approach is to click the **Activate Reference Character** button in the Main Toolbar. This button activates and selects the corresponding character.
2. Select **File > Import**, and browse to select the **HallTableBroken.obj** file. Do not modify any of the default import options. Click **OK**.
3. If you examine the graphical object carefully, you will see it is made up of three distinct parts that are joined along **break lines**. These indicate the three pieces of the table after it collapses following a collision.

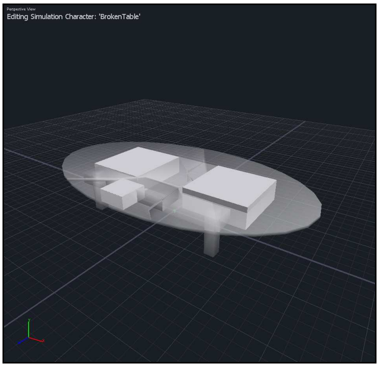


To reshape the mass objects:

Before adding new collision objects, it is important to adjust the shape, size and position of the three mass objects so that they roughly reflect the size of their corresponding table component.

1. Select the **Table01Mass** mass object.
2. Use the Property Editor to change its geometry type to **Cuboid**.
3. Use the Move, Rotate and Scale tools to adjust the position and size of this mass object. The precise shape and size of this mass object does not matter. Your main aim is to ensure that the mass object is located at the centre of mass of the part of the table it represents, and has the right volume to reflect the mass of that table part.

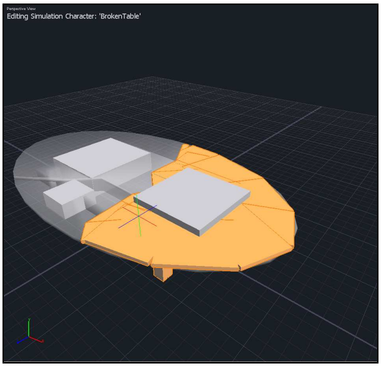
4. Repeat this process for the other two mass objects. You should end up with three mass objects that are positioned roughly according to the screenshot below:



To fill out the prop with collision objects:

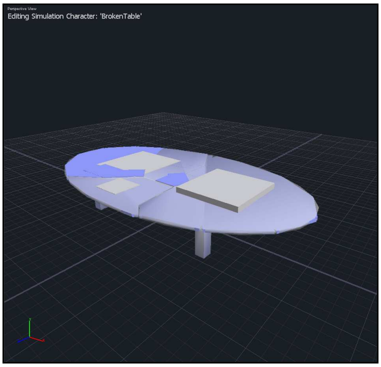
You can now start adding collision objects to fill out the three parts of the table. Before you create each collision object, select the corresponding mass object.

1. Select the **Table01Mass** object.
2. Click the **Create Box Collision Object** button on the Main Toolbar.
3. Use the Move, Rotate and Scale tools to adjust the size and position of this collision object. This collision object will form part of the table surface.
4. Continue creating collision objects that are children of the **Table01Mass** object until you have roughly represented the shape of the first section of the table surface. Keep in mind that you can penetrate collision objects that belong to the **same** mass object.



5. Create collision objects to fill out the other two sections of the table surface. You will need to create collision objects of the **Table02Mass** and **Table03Mass** objects respectively.

   When you are creating these collision objects, keep in mind that you **cannot** allow collision objects belonging to **different** mass objects to penetrate. If this occurs, then when the object is simulated, you will find that the different sections of the table will rapidly spring apart as the penetrating collision objects attempt to separate themselves.



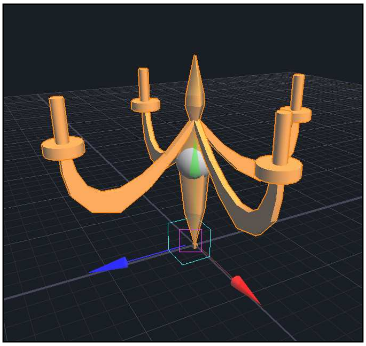
6. When you have finished filling out the three sections of the table, select **File > Save Character**, and name it **BrokenTable**. Click **OK**.

Step 3: Creating a chandelier

In this step we will build a simple rigid prop character that represents a **chandelier**. This will be a simple, rigid prop character based upon the Standard Prop Character. It will be made up of a single joint and a single mass object.

To create a chandelier:

1. Select **File > New Character…**, and select the **Standard Prop Character**. Click **OK**.
2. Select the **rootMass** object.
3. Select **File > Import…,** and browse to select the **Resources\Tutorials\Tutorial 6 – Creating Props\ HallChandelierBody.obj**. Do not modify the default Import Options. Click **OK**. A graphical object in the shape of a chandelier is created.
4. Select the graphical object, and use the Move tool to move it so that the **rootMass** object is located roughly at the centre of mass of the chandelier.



To reshape the mass object:

The mass object needs to be reshaped to better match the mass distribution of the chandelier.

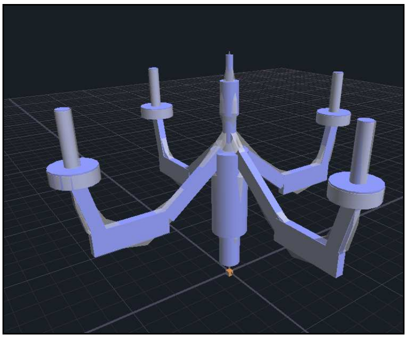
1. Select the **rootMass** object.
2. Use the Property Editor to change its **Type** property to **Cylinder**. Also, set its **Radius** property to 0.7, and its **Length** property to 1.5.
3. Use the Move and Rotate tools, reposition the cylindrical mass object so that its long axis coincides with the axis of the chandelier.

4. The mass object now has a mass distribution that more closely resembles the distribution of mass in the chandelier. However, the overall mass of the object—which depends on its volume and density—is now unrealistically large. Use the Property Editor to reduce its **Density** property to from 0.9 to 0.3. This will reduce the overall mass of the chandelier by two-thirds, which will make it more responsive to forces.

Note that when you move and rotate a mass object using the Move and Rotate tools, any child collision graphical objects which belong to it will move and rotate with it while maintaining their position relative to it. If you scale the parent mass object using the Scale tool however, any child mass objects will remain unaffected.

To fill out the prop with collision objects:

- As in previous examples, the chandelier graphical object needs to be filled out with collision objects. Keep in mind that you must preselect the **rootMass** object before creating each new collision object.
- You may want to hide the **rootMass** object while you are adding collision objects. When mass objects are large, they can sometimes obscure their child collision objects. To hide this mass object, click on its icon in the Node View. This method hides the specific mass object selected. Alternatively, **right-click** in the viewport and turn off **Mass Objects**. This method hides all mass objects in the scene using a global filter, regardless of whether their individual visibilities are turned on or off.



- When you have finished filling out the chandelier graphical object, select **File > Save Character**, and save this prop character as **MyChandelier**. Click **OK**.

Step 4: Adding a chandelier chain to a scene

In this step we will create a new scene and add the **MyChandelier** prop character to the scene.

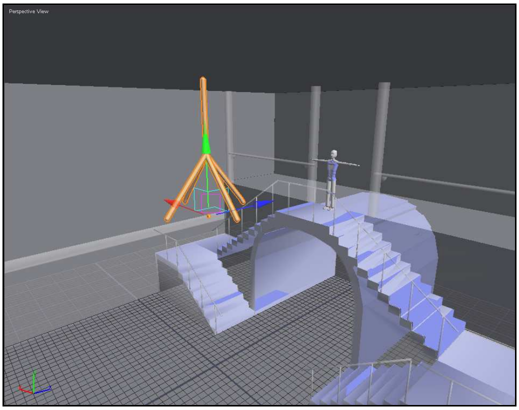
You will then use a **constraint** event to temporarily connect the chandelier to a set of objects representing a chain holding the chandelier to a virtual ceiling. Constraints are very useful when you want to temporarily connect characters—or parts of characters—together. By carefully choosing when to apply and release constraints, you can generate many complex physical effects. This technique was first described in Tutorial 4.

To add a chandelier chain to a scene:

1. Select **File > Open Scene**, and open the **Resources\Tutorials\Tutorial 6 - Creating Props\Tutorial 6 - Creating Props - Begin.ens** scene. This is a copy of the scene that you worked on in Tutorial 4 – Creating Environments.
2. Delete all timeline events in the Timeline Editor.
3. Select **File > Import**, and browse to select the **HallChandelierChain.fbx** file. from the folder. Turn off the **Create Animation Event** and **Create Simulation Events** settings. Click **OK**.
4. The **HallChandelierChain.fbx** file contains a character in the form of an articulated chain. When this file is imported into *endorphin*, a new prop character is created directly in the scene.

   You can create prop characters **directly** in scenes by importing FBX, BVH, XSI, ASF or Vicon VSK files directly, as an alternative to using Character Edit Mode. However, the prop character will not have been saved out as an **.nmc** character file, and so cannot be used in other scenes. However, this approach is useful when you do not need to reuse props in other scenes.

   If you **do** decide to reuse a prop character that you have created by directly importing a file, you can select that character in the scene, and then select **File > Export** and save the character as an **.nmc** *endorphin* character file.
5. Select the **HallChandelierChain** character in the Timeline Editor.
6. Use the Move tool to move the chain character until the topmost point of the chain coincides with the ceiling.

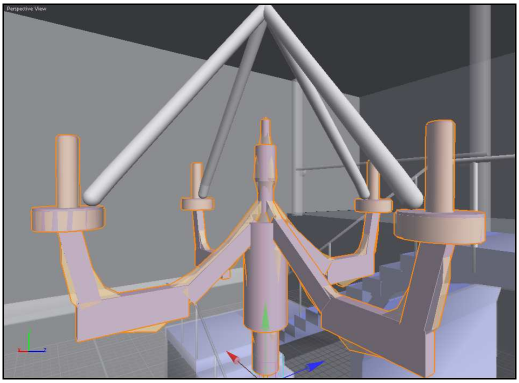


Step 5: Connecting a chandelier to the chain

In this step, we will add an instance of the **chandelier** prop character to the scene, and then use a constraint to connect it to the **chain** prop character that you created in the previous step.

To add a chandelier prop to the scene:

1. Select **Character > Add Character**.
2. Browse to the **Resources\Tutorials\Tutorial 6 - Creating Props** folder.
3. You should find that the MyChandelier prop character now appears in the **Add Character** dialog. Select this character and click **OK**.
4. Use the **Move** tool to move the new chandelier prop character up so that it sits just under the chain prop.



To connect the chandelier to the chain:

1. **Right-click** on the **MyChandelier** character timeline in the Timeline Editor, and select **Create Constraint Event**. This will add a new constraint event to the scene, with the chandelier **rootMass** mass object as the constrained mass object.
2. In **Selection** mode, hold down the **Ctrl** key, and select the four end-most mass objects of the **HallChandelierChain_fbx** character. This adds these mass objects to the set of constrained mass objects.
3. Exit Selection mode by **right-clicking** in the viewport, or by pressing the **Esc** key.
4. Ensure that the new constraint event is selected.
5. Use the Property Editor to set its **Start Frame** to Frame 0, and its **End Frame** to Frame 500.
6. Use the Property Editor to set its **Type** to JoinBodies.

   The default constraint type is Lock, which locks the position and orientation of the selected mass objects in **global** coordinates. The JoinBodies constraint type does not enforce any global position or orientation. Rather, it temporarily constrains the selected mass objects to maintain their **relative** positions and orientations for the duration of the event.

To connect the chandelier chain to the environment:

1. **Right-click** on the **HallChandelierChain** character timeline in the Timeline Editor, and select **Create Constraint Event**. This will add a new constraint event to the scene, with the chain **joint1Mass** mass object as the constrained mass object. This is the topmost mass object.
2. Use the Property Editor to set its **Start Frame** to Frame 0, and its **End Frame** to Frame 500.

To swing the chandelier:

The chain has been constrained so that its topmost mass object has its position and orientation locked in global coordinates. Also, the chandelier has been constrained so that it is joined to the mass objects at the lower ends of the chain.

1. **Simulate** the scene. You should find that the chandelier will slightly drop under the influence of gravity until the chain is stretched to its limit.

   The chain is made up of a series of short joints. This is a useful technique when you want to model chains, cables and other flexible entities. To test the flexibility of the chain, apply a horizontal force to the chandelier to set it swinging.

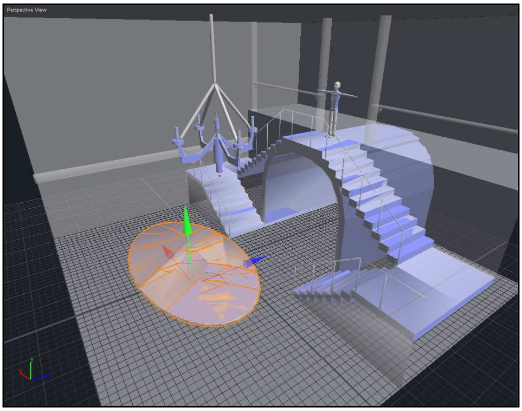
2. **Right-click** on the **MyChandelier** character timeline, and select **Create Force Event**.
3. With the force event selected, use the Property Editor to set its **Strength** to 3.0.
4. **Simulate** the scene again. The chandelier should now swing as a result of this force. You may find that the chain wobbles as it swings.
5. You can slightly **stiffen** the chain by adding an active pose event to the chain prop character. **Right-click** on the **HallChandelierChain** character, and select **Create Active Pose Event**.
6. With the active pose event selected, use the Property Editor to set its **Start Frame** to Frame 0 and its **End Frame** to Frame 500. If you simulate again, you should find that the chain appears slightly stiffer and the wobbling is reduced. To enhance or reduce this effect, modify the **Strength** property of the active pose event.
7. **Delete** the force event. We added the force to test the flexibility of the chain. The force is not required for the intended final scene simulation.

Step 6: Simulating the scene

In this step, we will add instances of the **Table** and **Chair** prop characters to the scene created in the previous step. We will also add an instance of the standard simulation character to the scene, and have this character interact with the table, chair and chandelier.

To add a table to the scene:

1. Select **Character > Add Character,** and select the **BrokenTable** character. Click **OK**.
2. Select this character, and use the Move tool to position it near the staircase, under the chandelier. See the screenshot below for guidance.



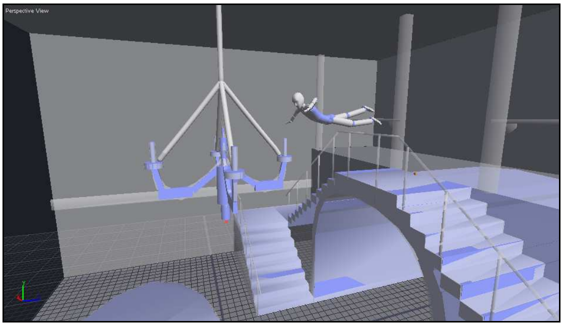
To add some chairs to the scene:

1. Select **Character > Add Character**, and select the **Chair** character. Click **OK**.
2. Select this character, and use the Move and Rotate tools to position it so that it is sitting next to the table.
3. Add a second instance of the Chair prop character, and use the Move and Rotate tools to position it so that it is sitting on the opposite side of the table to the first chair.

To simulate the character jumping towards the chandelier:

1. The scene already has an instance of the standard simulation character standing on the top of the platform. **Select** this character by clicking the **Character01** in the Timeline Editor.
2. Use the **Rotate** tool to rotate the character so that it is facing the chandelier.
3. Use the **Move** tool to move the character slightly closer to the edge of the platform.
4. **Right-click** on the **Character01** timeline track and select **Create Behaviour Event**. This creates a new behaviour event for this character.
5. Use the Property Editor to set the behaviour event **Name** to **Jump And Dive**. Also, set the **Start Frame** to Frame 0.
6. Delete the **Prop0**, **Prop01**, **Prop02** and **Prop03** mass objects that make up part of the railing in front of the character. They are not required for this simulation.

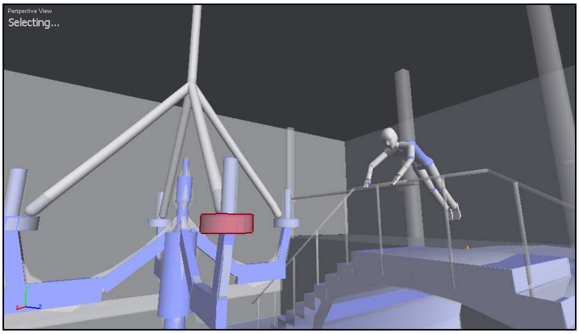
7. **Simulate** the scene. The character should now jump forward towards the chandelier, and should collide with it.



To simulate the character reaching out for the chandelier:

This scene will be appear more realistic if the character reaches out towards the chandelier.

1. Select the **Jump And Dive** behaviour event, and use the Property Editor to set its **End Frame** to Frame 75.
2. Create another behaviour event on the same **Character01** timeline.
3. Set the **Name** of this behaviour event to **Hands Reach And Look At**. Also, set its **Start Frame** to Frame 75.
4. Locate the **[Select]** command hotlink in the Property Editor that is associated with the **LookAtTarget**. Click this command hotlink to enter **Selection** mode, and select the **Cylinder11** collision object. Repeat this process for the **LeftHandTarget** property.



5. **Simulate** the scene again. When the character jumps, it should now reach out for the chandelier with its left hand. Also, its head should point towards the chandelier as it jumps.

To simulate the character catching the chandelier:

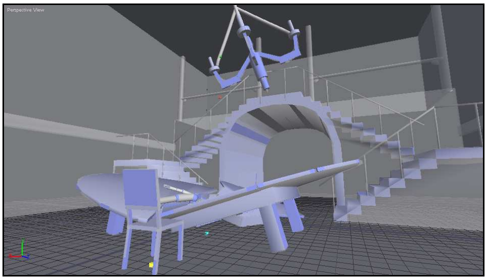
Our next goal during this simulation is to ensure that the character successfully **catches** the chandelier, rather than simply colliding with it.

1. **Right-click** on the **Character01** timeline, and select **Create Constraint Event** to create a new constraint. This should be roughly around Frame 110.
2. Use the **Time Slider** to identify the frame at which the character's hand collides with the chandelier. Set its **Start Frame** of the new constraint event to match this frame.
3. Select the constraint, and use the Property Editor to set its **Type** to JoinBodies.
4. Click the **[Select]** command hotlink that is found next to the Target Objects property in the Property Editor. This places you in **Selection** mode.
5. In Selection mode, select the **LeftHandMass** mass object of Character01. Hold down the **Ctrl** key to select additional objects. Select any object in the chandelier prop. This will add the **rootMass** object to the constraint event target object set.
6. **Simulate** again. The character should now jump and hold on to the chandelier, and then swing underneath it. The character remains attached to the chandelier due to the effect of the constraint event. When the constraint event finishes, the character falls and collides with the table and chairs underneath the chandelier. Constraint events are useful when you want to mimic the effect of a character holding onto an object.
7. To make the character **writhe** as it holds onto the chandelier, adjust the **End Frame** property of the **Hands Reach And Look At** behaviour so that it matches the **Start Frame** of the constraint event that connects the character's left hand to the chandelier. Then, add a new behaviour to the **Character01** timeline, and set its Name property to **Writhe**.
8. **Simulate** again. The character should now writhe as it dangles under the chandelier.

To allow the table to collapse:

When the character collides with the table at the end of this simulation, we want the table to collapse. To do so, we need to **sever** the joints that are holding the table prop character together.

1. Use the Time Slider to identify the frame at which the character first collides with the table. This should be roughly around Frame 200.
2. **Right-click** on the **BrokenTable_ref** character timeline, and select **Create Sever Event**.
3. Adjust the **Frame** of the sever event so that it coincides with the character striking the table.
4. Locate the **[Select]** command hotlink associated with the **Target Object** property in the Property Editor. Click this command hotlink to enter **Selection** mode.
5. In Selection mode, hold down the **Ctrl** key and select each of the table sections. The corresponding mass objects are added to the sever event **Target Objects** list.
6. To exit selection mode, **right-click** in the viewport, or press the **Esc** key.
7. **Simulate** again. The table will now break into three separate sections when the character collides with it. This is because the sever event detaches each of the parent joints of the mass objects in the **Target Objects** list.



Experimenting with this scene

There are different ways you can modify the flow of action in this scene. For example:

- Try experimenting with different **behaviours** on the simulation character to get different results.
- Try changing the duration of the **constraint** that joins the simulation character to the chandelier.
- Try adding some **helper forces** to the chairs at the frame in which the simulation character collides with the table. Helper forces can help create more dramatic action.

Conclusion

In this tutorial, you have learned how to create more complex prop characters using different techniques. You have also used a range of different event types to create dynamic interaction between a simulation character and various environment objects and prop characters.

If you have any problems with this tutorial, open the corresponding scene **Resources\Tutorials\Tutorial 6 – Creating Props\Tutorial 6 – Creating Props - Complete.ens**. This scene is an example of how your scene should look if you have successfully followed all the steps in this tutorial.

Tutorial 7

Importing and modifying animation

In the previous tutorials, you have created characters and scenes, and simulated those scenes using a variety of *endorphin* timeline event types. Typically, each simulation has started with the character in a relatively static standing position.

In this tutorial you will add **imported animation** into some parts of your simulation. You can use animation data at any point during a simulation. A common use of animated data is to **start** a simulation with characters driven by animation, and then **transition** to dynamic simulation when you want *endorphin* to begin generating motion. You can also blend back into animate data.

This tutorial will introduce the process of importing animated data, and transitioning from animation to simulation. The following tutorial, **Tutorial 8 – Blending From Simulation To Animation**, will cover the slightly more complex process of transitioning back from simulation to animation.

Before you begin

Before we begin this tutorial, it is important to understand some key concepts associated with *endorphin* simulations and animation.

Introducing simulation modes

All characters in an *endorphin* scene can be either **simulated** or **animated**. This is true for all characters, including simulation characters, prop characters and the Environment character. At any given frame, a given character will have a specific **Simulation Mode**:

- The default simulation mode is **Full Simulation**. In Full Simulation mode, characters are influenced by gravity, as well as forces, constraints, active poses and sever events—and behaviour events, in the case of simulation characters. This mode is often simply called the **simulation mode**.

- You can also set the Simulation Mode to **No Simulation**, **Collision Only** or **Collision With Momentum**. These modes are all **animation modes**. When a character is in one of the animation modes, it is driven by animation **keyframes**, rather than by the simulation. Characters store animation keyframes in the form of **animation events** on their character track in the Timeline Editor.

Comparing simulation modes

When characters are in one of the **animation modes**, they are not influenced by gravity or collisions, or timeline events such as forces, constraints, active poses, behaviours or sever events. They are only driven by animation keyframes stored in animation events, or keyframes stored directly on their timeline track.

In contrast, when characters are in the **simulation mode**, they are not influenced by animation keyframes. They are only affected by physical effects such as gravity and collisions, and timeline events such as forces, constraints, active poses, behaviours and sever events.

Changing the simulation mode

To change the simulation mode of a character, you must use **simulation events**. Simulation events are single-frame events that specify the new simulation mode that is to begin on that frame. To add a simulation event to a character, select the character and then select **Character > Create Simulation Event**. Alternatively, **right-click** on the character track in the Timeline Editor and select **Create Simulation Event**.

Displaying the simulation mode

When a character is in one of the **animation** modes, its mass objects, collision objects, graphical objects and joints are all displayed in a **dark grey** colour. Each of the three animation modes uses a slightly different shade of this colour. In addition, the Timeline Editor background colour for that character track is displayed using a **light blue** colour. Once again, the three animation modes use slightly different shades of this colour.

In contrast, when characters are in the **simulation** mode, their various component objects are displayed in their standard colours. Also, their corresponding character tracks in the Timeline Editor are displayed in the standard Timeline Editor colouring.

The use of different colours in the viewport to distinguish animation modes from the simulation mode is very useful when you are creating and reviewing simulations—you can quickly identify the frames in which characters have transitioned from animation to simulation, and vice versa, without needing to consult the Timeline Editor.

Step 1: Preparing the scene

In this step, we will prepare a simple scene by adding a simulation character that has been reshaped to match the character used by an FBX animation file that will be imported in Step 2.

To prepare the scene:

1. Launch *endorphin*.
2. Select **Character01** and delete it.
3. Select **Character > Add Character**, and browse to the folder **Resources\Tutorials\Tutorial 7 - Importing and Modifying Animation**.
4. The character **AudioMotion2** to should appear in the character list. Select this character and click **OK**.

   This character is an *endorphin* simulation character that has been derived from the standard simulation character. It has been slightly reshaped to match the skeleton used to create various animation cycles that are available as FBX files in the **Resources\Animation\AudioMotion2\FBX** folder. When you import animation data from any of these files into the **AudioMotion2** character using dynamic motion mapping, you will use its corresponding reference character, **AudioMotion2_ref.nmc**, which is also found in the **Resources\Tutorials\Tutorial 7 - Importing and Modifying Animation** folder.

Step 2: Importing an animated run cycle

In this step, we will import animation data onto the AudioMotion2 character using dynamic motion mapping.

To import an animated run cycle:

1. Select the **AudioMotion2** character using the Timeline Editor.
2. Select **File > Import**, and browse to select the **Resources\Tutorials\Tutorial 7 - Importing and Modifying Animation\Run With Gun.fbx** file. Click **Open**.

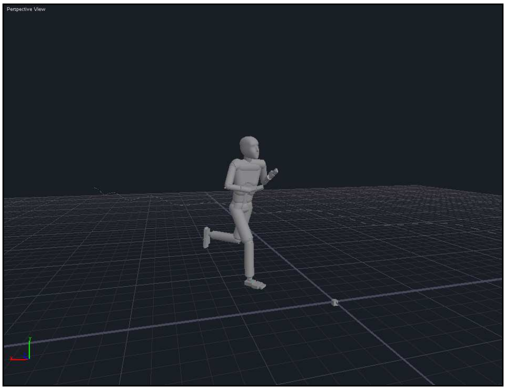
3. In the Import Options node hierarchy tree, select the **Hips** node. This ensures that only motion in the FBX file associated with the Hips joint, and any of its child joints, is imported onto the AudioMotion2 character.

4. Turn on the **Transfer From Reference Character** setting, and click the **Browse** button. Select the **AudioMotion2** character from the character list. Keep the other settings at their default values, and click **OK**.

5. The motion in the FBX file is imported, and a corresponding **animation event** is created on the AudioMotion2 timeline. This animation event contains the keyframed data. Ensure that the **Start Frame** of this animation event is Frame 0.

6. **Simulate** the scene. You should find that the character will now be driven by animation data. You will notice that the character is displayed in **dark grey**, rather than its usual light grey and blue colouring for mass objects and collision objects. This colour indicates that the character is being driven by animation data, and is not being simulated.

7. If you continue simulating past the End Frame of the animation event, the character will transition into **Full Simulation** mode, and will be displayed using its usual simulation colours. The character will fall to the ground under the influence of gravity.



8. If you examine the character timeline, you will notice that in addition to the animation event, two **simulation events** were automatically added at the start and end frames of the animation event. The first simulation event sets the simulation mode to **No Simulation**, and the second simulation event sets the simulation mode to **Full Simulation**. These simulation events control the frame range over which the character will be driven by animated data.

   In the Import Options dialog, you can specify whether or not to create simulation events when you import animation.

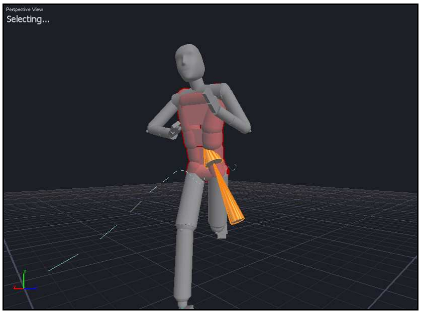
9. Move the second simulation event to an earlier frame, such as Frame 95, and simulate again. You should find that the character will be driven by the animation data until Frame 95, at which point the character will become simulated, and fall. Keyframes in the animation event **after** Frame 95 are ignored by the character, because it is in Full Simulation mode again.

Step 3: Combining animation with simulation

In this step, we want to extend the simulation to model the effect of an explosive blast. While the character is running, we will apply a strong force to the character so that it is thrown backwards violently.

To add a force to the animated character:

1. Move the Timeline Editor time slider so that the active frame is Frame 95.
2. Right-click on the character timeline and select **Create Force Event**. This adds a force event to the character timeline, and automatically enters Selection mode.
3. Hold down the **Ctrl** key, and select all the mass objects that form the pelvis and spine of the character. This updates the **Target Objects** property of the force in the Property Editor. To exit Selection mode, right-click in the viewport or press the **Esc** key.
4. Select the force event, and use the Property Editor to sets its **Strength** property to 30.0 and its **Rotation** property to –3.0.
5. **Simulate** the scene. You should find that the character transitions from animation to simulation at Frame 95, and is immediately thrown some distance by the force. You can change the strength and direction of the force to generate different responses. You can also change the target object set of the force. Make sure that the force is applied at the same frame—or within a few frames—of the simulation event. Do not set the force to a frame prior to the simulation event, otherwise the force will have no effect.



To make the character writhe in response to the force:

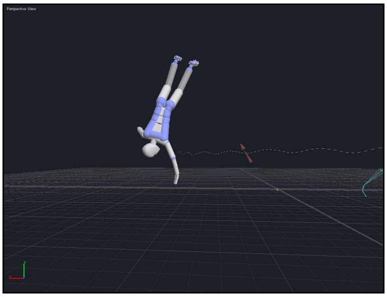
When you apply a strong force to a character, you may find that it is propelled in a ragdoll-like manner. That is, the character may appear to have a limp, lifeless response to the force. Applying some additional **behaviours** can improve the quality of the simulation.

1. **Right-click** the character track and select **Create Behaviour Event**.
2. Set the **Name** of the behaviour to **Writhe**.
3. Set the **Start Frame** of the behaviour to Frame 95.
4. Set the **End Frame** of the behaviour event to Frame 140.
5. Set the **Velocity** property of the behaviour to –0.8.
6. **Simulate** the scene. You should find the character now writhes as it moves through the air.

To make the character anticipate the ground:

As the character falls towards the ground, you can add a behaviour to make it appear that the character is anticipating the collision.

1. **Right-click** the character track and select **Create Behaviour Event**.
2. Set the **Name** of the behaviour to **Catch Fall**.
3. Set the **Start Frame** of the behaviour to Frame 140.
4. Set the **End Frame** of the behaviour to Frame 166.
5. Turn off the **Predict Ground Height** property.
6. **Simulate** the scene. You should find the character now reaches out as it approaches the ground.



To make the character tuck as it hits the ground:

As the character hits the ground, you can add a behaviour to make the character tuck into a foetal position. This is a useful way for making characters appear more lifelike during collisions.

1. **Right-click** the character track and select **Create Behaviour Event**.
2. Set the **Name** of the behaviour to **Body Foetal**.
3. Set the **Start Frame** of the behaviour to Frame 166.
4. **Simulate** the scene. You should find the character now tucks itself into a foetal position as it collides with the ground.

Further experiments

- Try changing the timings and durations of the various behaviour events. Each combination of timings will generate new motion.
- Try changing the direction of the force event. Even small changes to the direction can generate entirely different resulting motions.

Conclusion

In this tutorial, you have learned how to import animation, and how to combine it with dynamic simulations to create more complex motion. You have also learned about simulation modes, and how to use simulation events to control the transitions between simulation and animation modes.

If you have any problems with this tutorial, open the corresponding scene Resources\Tutorials\Tutorial 7 – Importing And Modifying Animation\Tutorial 7 – Importing And Modifying Animation - Complete.ens. This scene is an example of how your scene should look if you have successfully followed all the steps in this tutorial.

Tutorial 8

Blending from animation to simulation

In the previous tutorial, you imported animation data and used **simulation events** to change the simulation mode from animation to simulation.

In this tutorial, you will use **transition events** to blend back from simulation mode to an animation mode. Transition events are similar to simulation events, except that they have a duration of many frames, rather than a single frame.

Before you begin

Before we begin this tutorial, it is important to understand some key concepts associated with *endorphin* simulations and transition events.

Using simulation and transition events

You will typically use **simulation** events whenever you want to change the mode from animation to simulation, and **transition** events whenever you want to change the mode back to animation.

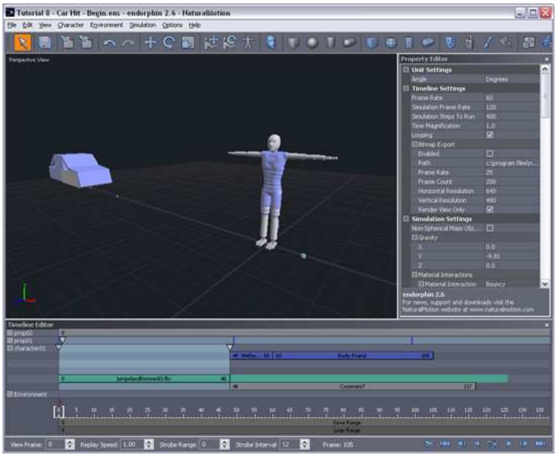
The reason is that it is easy for *endorphin* to begin dynamically simulating an animated character without any discontinuities in its position or motion. In contrast, blending back from simulation to animation is more difficult to achieve in a realistic manner. This process requires a transition over a number of frames in order to guarantee a smooth blend between the character's dynamically simulated position and momentum, and the position and momentum specified by the animation data.

Step 1: Loading the scene

In this step we will open an existing scene that we will use for experimenting with transition events.

To load the scene:

1. Launch *endorphin*.
2. Select **File > Open Scene…** to browse to select **Resources\Tutorials\Tutorial 8 - Blending from Simulation to Animation\Car_Hit_Begin.ens**.



Tutorial 8 - Car Hit - Begin.ens - endorphin 2.6 - NaturalMotion
File Edit View Character Environment Simulation Options Help
Perspective View
Property Editor
Timeline Editor
prop02
prop01
character01
Environment
Save Range
Loop Range

Property Editor

Unit Settings	
Angle	Degrees
Timeline Settings	
Frame Rate	60
Simulation Frame Rate	120
Simulation Steps To Run	400
Time Magnification	1.0
Looping	✓
Bitmap Export	
Enabled	☐
Path	c:\program files\n…
Frame Rate	25
Frame Count	200
Horizontal Resolution	640
Vertical Resolution	480
Render View Only	✓
Simulation Settings	
Non-Spherical Mass Obj…	☐
Gravity	
X	0.0
Y	-9.81
Z	0.0
Material Interactions	
Material Interaction	Bouncy

endorphin 2.6
For news, support and downloads visit the NaturalMotion website at www.naturalmotion.com

View Frame: 0 Replay Speed: 1.00 Strobe Range: 0 Strobe Interval: 12 Frame: 105

3. This scene contains a simulation character, along with a **car** prop character that represents a simple vehicle shape. The car prop has been animated with a few keyframes so that it moves towards the simulation character. The simulation character has an animation event containing a **run-and-jump** cycle.
4. **Simulate** the scene. The car prop moves towards the character, controlled by its keyframes. Between Frame 0 and Frame 48, the simulation character is driven by its animation event. You can tell that the character is animated, rather than simulated, by the fact that it is displayed in the **dark grey** animation colour. At Frame 48, the character and car collide, and a simulation event has been added to begin dynamically simulating the character. The character writhes and tucks as a result of the **Writhe** and **Body Foetal** behaviours that have been added to it.

Step 2: Using helper props and constraints

In this step we will explore the scene opened in Step 1 in more detail.

Animating using helper props

If you examine the Timeline Editor, you will note that there are **two** prop characters—**prop01** and **prop02**. The car prop itself is **prop02**. The keyframes have been applied to **prop01**, which is a small **helper** prop connected to the prop02 via a constraint.

This technique—animating a prop character indirectly using a helper prop connected by a constraint—is a useful *endorphin* modeling technique. If we had animated the car **directly**, it would be in an **animation** mode. It would be able to collide with other scene objects and transfer momentum to them, but its motion would be completely unaffected by collisions.

In contrast, by animating a helper prop and constraining the car to this helper, we can keep the car prop in **simulation** mode. When the car collides with objects, it can react to collisions dynamically. For example, it may slightly bounce or wobble during collisions. You can vary the strength of these **secondary dynamics** by varying the strength of the constraint. Usually these effects will be quite subtle but can add to the realism of a scene.

Using helper constraints to control dynamic motion

If you examine the Timeline Editor, you will note that the simulation character has a **constraint** that applies between Frames 48 and 117. This is the period in which the simulation character is spinning through the air. The constraint only locks the **rotation** of the character about the global **X-axis**.

We have added this **helper constraint** to help ensure that as the character spins through the air, its roll axis stays aligned to the global X-axis. Constraints are often useful when you want to add some control to what would otherwise be unconstrained dynamic motion.

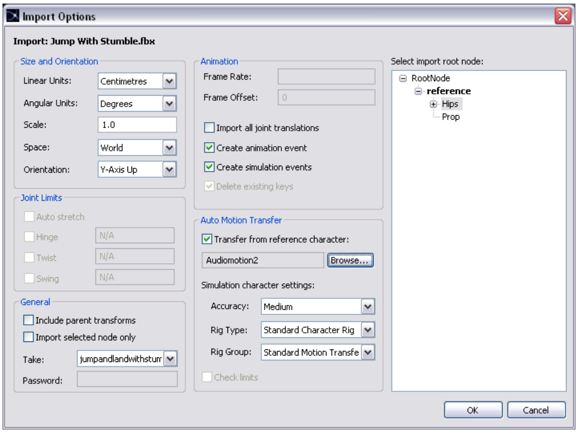
Step 3: Importing blend animation

When you simulate the current scene, the character runs at the car, collides with it and ends up tucked into a ball on the ground behind the car.

In this step, we are going to create another animation event for the simulation character. We will import animated motion of the character landing from a jump with a stumble and run. Our goal is to blend from the dynamic simulation into this animation. In a later step we will add the transition event to blend from the simulation into the animation.

To import animated data representing the final blend animation:

1. Select **Character01**.
2. Select **File > Import**, and browse to select **Resources\Tutorials\Tutorial 8 - Blending from Simulation to Animation\Jump With Stumble.fbx**. Click **Open**.
3. In the **Import Dialog**, select the **Hips** node in the joint hierarchy tree.
4. Turn off the **Create Simulation Events** setting.
5. Turn on the **Transfer From Reference Character** setting, and browse to select the **Audiomotion2** reference character. Click **OK** to import this motion as a new animation event.


Import Options

Import: Jump With Stumble.fbx

Size and Orientation
Linear Units: Centimetres
Angular Units: Degrees
Scale: 1.0
Space: World
Orientation: Y-Axis Up

Joint Limits
☐ Auto stretch
☐ Hinge N/A
☐ Twist N/A
☐ Swing N/A

General
☐ Include parent transforms
☐ Import selected node only
Take: jumpandlandwithstur
Password:

Animation
Frame Rate:
Frame Offset: 0
☐ Import all joint translations
☑ Create animation event
☑ Create simulation events
☑ Delete existing keys

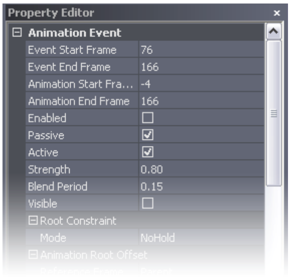
Auto Motion Transfer
☑ Transfer from reference character:
Audiomotion2 Browse...
Simulation character settings:
Accuracy: Medium
Rig Type: Standard Character Rig
Rig Group: Standard Motion Transfe
☐ Check limits

Select import root node:
RootNode
reference
Hips
Prop

OK Cancel

To adjust the animation event timing:

1. **Select** the new animation event in the Timeline Editor. You may want to move the animation event marker to a new timeline track.
2. Set the **Event Start Frame** to Frame 76.
3. Set the **Event End Frame** to Frame 166.
4. Set the **Animation Start Frame** to Frame -4.
5. Set the **Animation End Frame** to Frame 166.



Property Editor

Animation Event

Event Start Frame	76
Event End Frame	166
Animation Start Fra...	-4
Animation End Frame	166
Enabled	☐
Passive	☑
Active	☑
Strength	0.80
Blend Period	0.15
Visible	☐
Root Constraint	
Mode	NoHold
Animation Root Offset	
Reference Frame	Parent

Step 4: Repositioning the blend animation

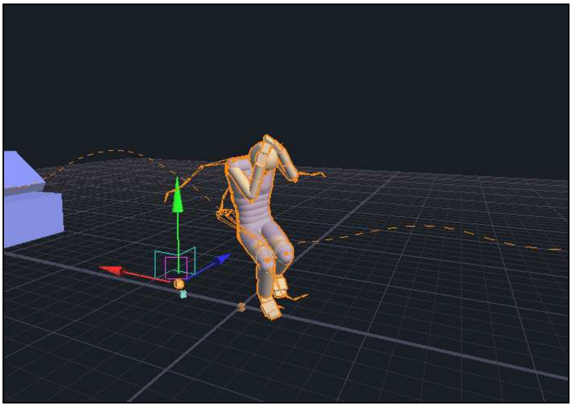
In this step, we will **reposition** the animation data so that it will blend smoothly with simulated motion.

When you imported animation data in the previous step, its position and orientation were initially based on the data in the FBX file. However, *endorphin* also lets you **move** and **rotate** animation data directly in the scene itself. This is very useful for repositioning the animation so that it coincides with the dynamically simulated motion of the character during the intended transition frames. It is very important to position animated data correctly to create realistic blends from simulation to animation.

To adjust the position of the animation event data:

1. Turn on animation event data visualisation. To do so, **select** the animation event. Use the Property Editor to turn on its **Visible** setting. You should see a green jointed skeleton displayed in the viewport, along with a dashed line that indicates the **motion path** of the animation data root.
2. **Select** the animation event. When you are visualising the animation data, you can select the green animation data skeleton or motion path as an alternative to selecting the animation data marker in the Timeline Editor.
3. **Simulate** to fill the frame buffer with at least 100 frames.
4. Move the time slider to Frame 97. This is the frame at which the character first makes contact with the ground.
5. Use the **Move** and **Rotate** tools to modify the global position and orientation of the animation. Your goal is to ensure that the animated character position closely matches the simulated character position. When you use these tools, you are modifying the **Animation Root Offset** property of the animation event.

If you use the Rotate tool, ensure that you only rotate about the global **Y-axis**. Do not rotate about the global X-axis or Z-axis, as this will have the unwanted effect of moving some parts of the animation cycle off the ground plane.



6. You should find that the following values for the position and orientation of the animation root offset work well:

    - **X Orientation:** +180.0.
    - **Y Orientation:** -51.9.
    - **Z Orientation:** +180.0.
    - **X Position:** 0.748.
    - **Y Position:** 0.0.
    - **Z Position:** 0.176.

Step 5: Transitioning into the blend animation

We have added the blend animation data, and repositioned it so that it will blend well with the motion of the simulated character. However, at the moment this blend animation is not yet used by the simulation character.

In this step, we will add a **transition event** to the simulation character timeline. This event will change the **simulation mode** of this character back from simulation to animation over a range of frames.

To add a transition event to the character:

1. **Right-click** on **Character01** in the Timeline Editor and select **Create Transition Event**.
2. Set the **Start Frame** of the new transition event to Frame 90.
3. Set the **End Frame** of the new transition event to Frame 112.
4. If you examine the Timeline Editor, you will notice that the Character01 timeline track has a gradually strengthening **blue** colour added to its background over the duration of the transition event. This is the **animation mode** colour, and indicates that the character is driven by animation data rather than by the dynamic simulation.
5. **Simulate** the scene. The dynamic motion of the character now blends smoothly into the animated motion. If you use the time slider to examine the transition event in detail, you will notice that during the course of the transition event, the motion of the character is generated by a **combination** of simulated and animated motion.

   Initially, the character is entirely driven by simulated motion. Over the course of the transition event, the animated motion has increasing influence. By the end of the transition event, the character is entirely driven by animated data. This transition is reflected by the **colour** of the character. During the course of the transition, the character blends towards the **dark grey** animated character colour.

Step 6: Introducing active animation events

In this step, we will use an **active animation event** to improve the quality of the transition from simulated motion to animated motion.

Active animation events

Until now, we have described animation events as applying only when the simulation mode is set to one of the **animation modes**. When characters are driven by animated data in an animation mode, the data completely drives the character.

However, this is no longer strictly true. You can now also use animation events to drive characters when they are in the **simulation mode**. When you use animation data in this way, you effectively drive the character using a succession of **active poses**. This technique is called **active animation**.

Active animation does **not** affect the global position or orientation of the character root. Rather, it drives the individual joint orientations of a character. Active animation acts as a physical input during a dynamic simulation. You can specify the relative strength of the active animation by modifying the animation event **Strength** property.

Using active animation events

You can enhance a blend from animation to simulation using active animation events. Suppose you have a character driven by dynamic simulation, and at a particular frame you introduce a transition event blend into an animation event. By default, you would set the **Start Frame** of the animation event to match the **Start Frame** of the transition event.

To use the active animation technique, you would move the **Start Frame** of the animation event back a number of frames, and turn on its **Active** setting. For example, if a character has a transition event between Frames 100 and 125, and an animation between Frames 80 to 300, the character will effectively have four simulation ranges:

- Before **Frame 80**, the character is dynamically simulated.
- Between **Frames 80 and 100**, the character is dynamically simulated, with the animation event driving joint orientations dynamically.
- Between **Frames 100 and 125**, the character is blended between its simulated motion and the animation event.
- Between **Frames 125 and 300**, the character is driven entirely by the animation event.

To improve the transition by using active animation:

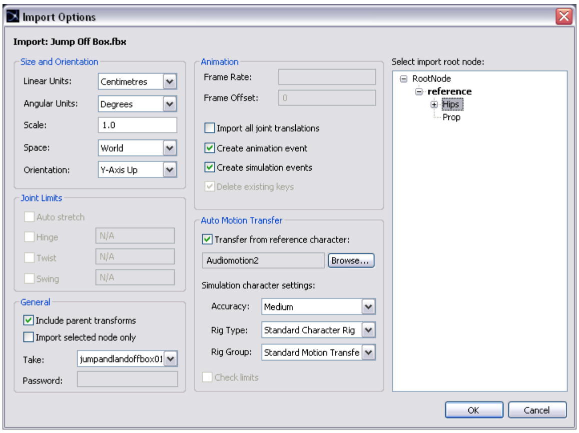
1. **Select** the animation event.
2. Turn on its **Active** property.
3. Set the **Strength** property to 0.8.
4. Set the **Blend Period** property to 0.15.
5. **Simulate** the scene. You should find that between Frame 76 and Frame 90, the animation event drives the character **actively**. That is, the character is still dynamically simulated, but its joints are influenced by the **Jump With Stumble** animation event. The active animation helps make the subsequent transition event between Frames 90 and 112 even smoother and more realistic.

Step 7: Experimenting with different blend animation

In this step, we will experiment with different scene variations by importing more animation data to create additional animation events. We will use the same transition event to blend to different animation events, in order to explore different scenarios.

To import more animation data:

1. Select **Character01** using the Timeline Editor.
2. Select **File > Import**, and browse to select the **Resources\Tutorials\Tutorial 8 - Blending from Simulation to Animation\ Jump Off Box.fbx**. Click **Open**.
3. In the **Import Options** dialog, select the **Hips** node from the node tree hierarchy.
4. Turn off the **Create Simulation Events** setting.
5. Turn on the **Transfer From Reference Character** setting, and browse to select the **Audiomotion2_ref** reference character. In required, you will need to browse to the **Resources\Characters\Custom** folder in order to locate the AudioMotion2_ref character. Click **OK**.


Import Options

Import: Jump Off Box.fbx

Size and Orientation
Linear Units: Centimetres
Angular Units: Degrees
Scale: 1.0
Space: World
Orientation: Y-Axis Up

Joint Limits
☐ Auto stretch
☐ Hinge N/A
☐ Twist N/A
☐ Swing N/A

General
☑ Include parent transforms
☐ Import selected node only
Take: jumpandlandoffbox01
Password:

Animation
Frame Rate:
Frame Offset: 0
☐ Import all joint translations
☑ Create animation event
☑ Create simulation events
☑ Delete existing keys

Auto Motion Transfer
☑ Transfer from reference character:
Audiomotion2 Browse...
Simulation character settings:
Accuracy: Medium
Rig Type: Standard Character Rig
Rig Group: Standard Motion Transfe
☐ Check limits

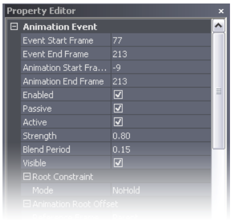
Select import root node:
RootNode
reference
Hips
Prop

OK Cancel

To adjust the animation event timing:

1. **Select** the new animation event in the Timeline Editor. You may want to move the animation event marker to a new timeline track.

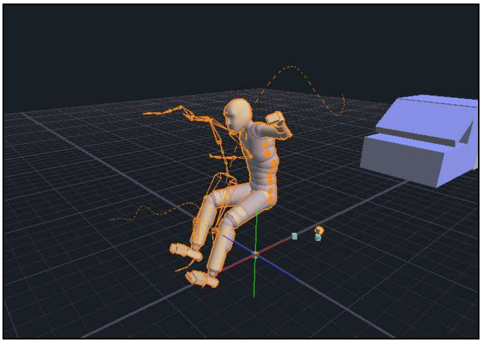
2. Set the **Event Start Frame** to Frame 77.
3. Set the **Event End Frame** to Frame 213.
4. Set the **Animation Start Frame** to Frame -9.
5. Set the **Animation End Frame** to Frame 213.


Property Editor

Animation Event	
Event Start Frame	77
Event End Frame	213
Animation Start Fra...	-9
Animation End Frame	213
Enabled	✓
Passive	✓
Active	✓
Strength	0.80
Blend Period	0.15
Visible	✓
Root Constraint	
Mode	NoHold
Animation Root Offset	

To adjust the position of the animation event data:

1. Simulate for at least 100 frames, and then use the time slider to set the active frame to Frame 97.
2. Turn on the **Visible** setting in the Property Editor.
3. Use the **Move** and **Rotate** tools to adjust the position and orientation of the animation data so that it coincides with the position of the simulated character at Frame 97.



4. You should find that the following values for the position and orientation of the animation root offset work well:
   - **X Orientation:** +180.0.
   - **Y Orientation:** -87.2.
   - **Z Orientation:** +180.0.
   - **X Position:** 0.882.
   - **Y Position:** 0.0.

- **Z Position:** 0.093.

To improve the transition by using active animation:

1. **Select** the animation event.
2. Turn on its **Active** property.
3. Set the **Strength** property to 0.8.
4. Set the **Blend Period** property to 0.15.
5. Right-click the **Jump With Stumble** animation event and select **Disable Event**.
6. **Simulate** the scene. You should find that the character motion now blends into the **Jump Off Box** animation.
7. Now that you have created two animation events, and positioned them both correctly, you can easily **toggle** between them by enabling one of the events, and disabling the other event. In a similar manner, to can import more animation to create more animation events for a given character; simply ensure that only one of the animation events is enabled at any one time.

Timeline priority

Each character has a number of timeline tracks. The topmost track has the **highest** priority, followed by the next track and so on, down to the bottom track. For any given frame, the animation event with the highest priority is used, and any other animation event at the same frame is ignored. This means that if you are experimenting with different animation events, you can modify their respective priorities as an alternative to enabling and disabling them. That is, if you want to use a particular animation event, move it so that it has the highest priority of any of its animation events.

Conclusion

In this tutorial you have imported animation data to create **animation events**, and visualised this data in the viewport. You have moved and rotated this animated data to obtain a good blend between simulation and animation, and then used **transition events** to actually generate the transition.

You have also experimented with **active animation events** to further improve your blends. Finally, you have experimented with different animation events in the same scene.

If you have any problems with this tutorial, open the corresponding scene **Resources\Tutorials\Tutorial 8 – Blending From Simulation To Animation\Tutorial 8 – Blending From Simulation To Animation - Complete.ens**. This scene is an example of how your scene should look if you have successfully followed all the steps in this tutorial.

Tutorial 9

Using multiple characters

In this tutorial, you will create a scene with **multiple** simulation characters.

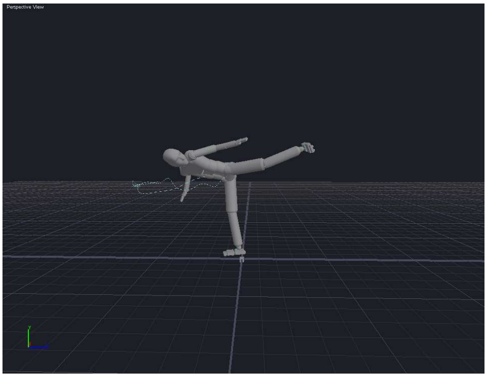
Each simulation character has its own set of force, constraint, sever, active pose, behaviour and animation events. By blending animation with simulation in a scene with multiple characters, you can generate more complex interactions such as tackles.

Step 1: Preparing a new scene with animation

In this step, we will create a new scene and populate it with the **AudioMotion2** simulation character. We will then import animation data onto this character. This animation will contain a **sidekick** motion.

To prepare the scene:

1. Launch *endorphin*, and delete the default character.
2. Select **Character > Add Character**, and browse to the **Resources\Tutorials\Tutorial 9 - Using Multiple Characters** folder. Select the **AudioMotion2** simulation character and click **OK**.
3. **Select** the new character in the Timeline Editor.
4. Select **File > Import**, and browse to select the **Resources\Tutorials\Tutorial 9 - Using Multiple Characters\Martial Idle To Side Kick.fbx** file. Click **Open**.
5. In the **Import Options** dialog, select the **Hips** node from the node hierarchy tree.
6. Turn on the **Transfer From Reference Character** setting, and browse to select the **AudioMotion2** reference character. Do not modify any other import setting. Click **OK**.
7. **Simulate** the scene. The character will be driven by animation data, and will be displayed in the **dark grey** animated mode colour.



Step 2: Adding momentum transfer

In this step, we will change the **simulation mode** property of the initial simulation event.

By default, the initial simulation event created when you import animation data will be set to the **No Simulation** mode. In this mode, the character cannot affect any other character in the scene. However, in this scene we would like an animated character to **kick** a simulated character. This requires **transferring momentum** from the kicking character to the other character. To do so, we need to change the simulation mode of the kicking character to **Collision With Momentum**.

To allow momentum transfer:

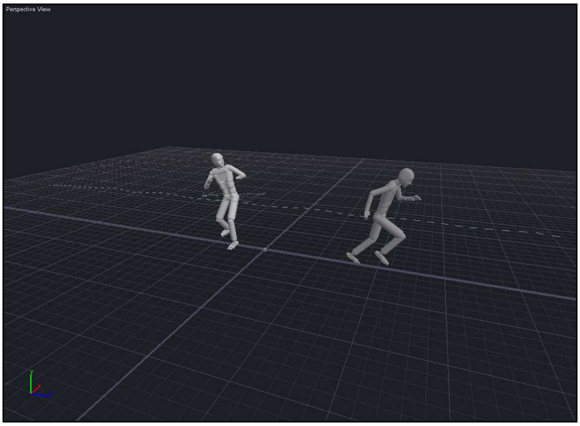
1. Select the **initial** simulation event . This is the event at Frame 0.
2. Using the Property Editor, set the **Simulation Mode** property to **Collision With Momentum**.

Step 3: Adding another character to the scene

In this step, we will add a **second** character to the scene—an instance of the **AudioMotion1** simulation character. We will then import animation data onto this character. This animation will contain a **run cycle**.

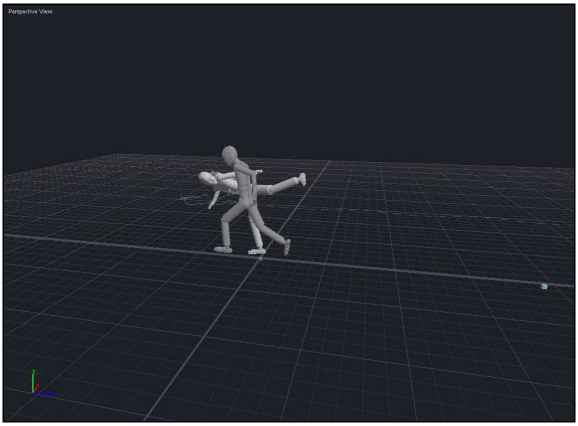
To add another character to the scene:

1. Select **Character > Add Character**, and select the **AudioMotion1** simulation character.
2. Select the **AudioMotion1** character in the Timeline Editor.
3. Select **File > Import**, and browse to select the **AM_Run.fbx** file.
4. Select the **Hips** node in the node hierarchy tree.
5. Turn on **Transfer From Reference Character** setting, and browse to select the **AudioMotion1** reference character. Do not modify any other import setting. Click **OK**. The animation data is imported as a new animation event on the **AudioMotion1** timeline. This animation drives the AudioMotion1 character in a **run cycle**.



To adjust the position of the animation event data:

1. Select the **AM_Run** animation event in **AudioMotion1** character timeline.
2. Using the Property Editor, modify the following **Animation Root Offset** properties:
   - **Y Orientation:** +180.0.
   - **X Position:** -0.068.
   - **Z Position:** +3.869.
3. **Simulate** the scene again. The **AudioMotion1** character should now run directly at the **AudioMotion2** character. However, the two characters will pass through each other. This is because neither character is in the **simulation mode**.



Step 4: Simulating a dynamic kick

In this step, we will simulate the running character a few frames before it collides with the kicking character.

At the moment of collision, the running character will be **simulated**, but the kicking character will be **animated**. However, the kicking character will be in **Collision With Momentum** mode, which means the kick will affect the physical dynamics of the running character.

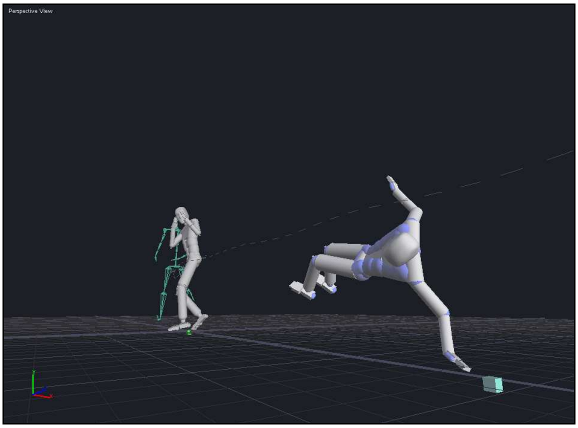
To simulate a dynamic kick:

1. Select the **second** simulation event on the **AudioMotion1** character timeline. This is the simulation event at the **end** of the animation event.
2. Set the **Frame** property of this event to Frame 96. This frame is a few frames before the impact between the two characters.
3. **Simulate** the scene. The running character will now receive the full impact of the kick.

To improve the kick response:

When simulation characters are hit by impact forces, it is important that their response appears lifelike. You should avoid limp, ragdoll responses. A good way to improve dynamic motion is to use behaviours.

1. **Right-click** on the AudioMotion1 character timeline, and select **Create Behaviour Event**.
2. Set the **Name** of this behaviour to **Writhe**.
3. Set the **Start Frame** of this behaviour to Frame 98.
4. Set the **End Frame** of this behaviour to Frame 126.
5. **Right-click** on the AudioMotion1 character timeline, and select **Create Behaviour Event**.
6. Set the **Name** of this behaviour to **Catch Fall**.
7. Set the **Start Frame** of this behaviour to Frame 126.
8. Set the **End Frame** of this behaviour to Frame 250.
9. **Simulate** the scene. The running character now writhes as it falls backward, and then reaches out to protect itself as it falls towards the ground.



Further experiments

- Try adjusting the **position** of the run cycle animation event to generate different types of collision.
- Try changing the **Momentum Transfer Factor** property of the simulation event. You can reduce this factor to soften the impact of the kick.
- Try changing the **behaviour timing** to change the way the running character responds to the kick.

Conclusion

You have created a scene using **multiple** characters, and used the **momentum transfer** technique to affect the motion of a simulated character using an animated character.

If you have any problems with this tutorial, open the corresponding scene **Resources\Tutorials\Tutorial 9 - Using Multiple Characters\Tutorial 9 – Multiple Characters - Complete.ens**. This scene is an example of how your scene should look if you have successfully followed all the steps in this tutorial.

Tutorial 10

Active posing

In the previous tutorials, you have used a range of different event types to influence dynamic motion. In particular, you have used forces, constraints, behaviours and sever events, as well as active animation events.

In this tutorial, you will work with another event type—**active pose** events. Active poses are conceptually similar to keyfames in a traditional animation system. An active pose represents a **desired** or **intended** character pose that influences the character's joints over a range of frames.

Depending on the duration and strength of the active pose, it may have a subtle effect or a more pronounced effect on the simulation. Keep in mind that active poses are one of many influences on the character's dynamic motion.

Step 1: Loading the scene

In this step, we will load a scene that is composed of a simulation character connected to gymnastic rings via a constraint. The simulation character represents a gymnast and has a more developed physique than the standard simulation character.

To load the scene:

1. Launch *endorphin*.
2. Select **File > Open Scene…**, and browse to select **Resources\Tutorials\Tutorial 10 - Active Posing\ Tutorial 10 - Active Posing - Begin.ens**. Click **Open**.
3. **Simulate** the scene. This scene is composed of a single simulation character connected to gymnastic rings via a constraint. Initially, the character will simply hang off the rings in a lifeless manner.

Step 2: Adding an active pose

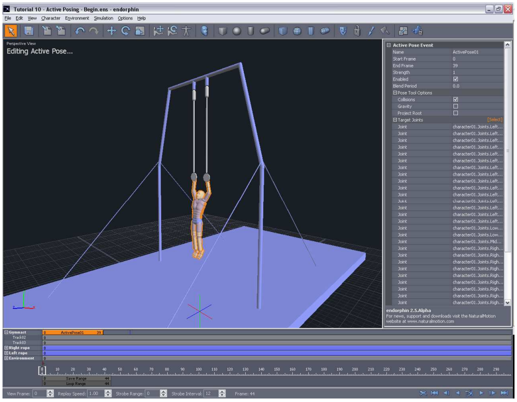
In this step, we will add an **active pose** to the character timeline. This pose will drive the relative joint motion of the simulation character.

This scene is a relatively contrived example in which **all** the character motion is generated via active poses. In most scenes, active poses are used to influence character motion in more subtle ways.

To add an active pose:

1. **Right-click** on the **Gymnast** timeline track in the Timeline Editor and select **Create Active Pose Event**.
    - You will be placed in **Editing Active Pose** mode. Whenever you create or select an active pose event, *endorphin* will automatically place you in **Editing Active Pose** mode, with the Pose Move tool activated.
    - In this mode, the character is displayed in the corresponding **active pose** for this event. You can use the **Pose Move** or **Pose Rotate** tools to edit the active pose. Alternatively, you can import an *endorphin* **.nma** pose file to set the active pose.
    - Active poses only affect **individual joint orientations**. An active pose does **not** affect the overall position or orientation of the character itself. You should not use the Move or Rotate tools to move or rotate the active pose, as these have **no effect** on the relative pose.
    - To exit Editing Active Pose mode, press the **Esc** key.
2. Exit **Editing Active Pose** mode.
3. Set the **Start Frame** of the active pose to Frame 0.
4. Set the **End Frame** of the active pose to Frame 39.
5. **Right-click** on the active pose event marker, and select **Load Pose…**.
6. Browse to select the file **Resources\Tutorials\Tutorial 10 - Active Posing\Pose 1.nma**. You should see the active pose in the viewport update to reflect the pose stored in the **Pose1.nma** pose file.
7. Set the **Strength** property of this active pose to 4.0. The strength of an active pose defines the relative influence of the active pose on the simulation.

8. **Simulate** the scene. You should find that the character now moves towards the active pose, and holds this poses until Frame 39. At Frame 39, the influence of the active pose ends and the character relaxes.


Tutorial 10 - Active Posing - Begin.ens - endorphin
File Edit View Character Environment Simulation Options Help
Perspective View
Editing Active Pose...
Gymnast
ActivePose01
Right rope
Left rope
Environment
endorphin 2.5.Alpha
For news, support and downloads visit the NaturalMotion website at www.naturalmotion.com

Active Pose Event	
Name	ActivePose01
Start Frame	0
End Frame	39
Strength	1
Enabled	✓
Blend Period	0.0
Pose Tool Options	
Collisions	✓
Gravity	☐
Project Root	☐
Target Joints	[Select]
Joint	character01.Joints.Left...
Joint	character01.Joints.Left...
Joint	character01.Joints.Left...
Joint	character01.Joints.Left...
Joint	character01.Joints.Left...
Joint	character01.Joints.Left...
Joint	character01.Joints.Left...
Joint	character01.Joints.Left...
Joint	character01.Joints.Left...
Joint	character01.Joints.Left...
Joint	character01.Joints.Left...
Joint	character01.Joints.Left...
Joint	character01.Joints.Left...
Joint	character01.Joints.Left...
Joint	character01.Joints.Left...
Joint	character01.Joints.Low...
Joint	character01.Joints.Low...
Joint	character01.Joints.Mid...
Joint	character01.Joints.Righ...
Joint	character01.Joints.Righ...
Joint	character01.Joints.Righ...
Joint	character01.Joints.Righ...
Joint	character01.Joints.Righ...
Joint	character01.Joints.Righ...
Joint	character01.Joints.Righ...
Joint	character01.Joints.Righ...
Joint	character01.Joints.Righ...

Step 3: Adding more active poses

In this step, we will add three more **active poses** to the character timeline. These poses will continue drive the relative joint motion of the simulation character. These active poses create the first half of a gymnastic movement.

To add a second active pose:

Repeat the previous steps to add another active pose event to the Gymnast character timeline:

1. Load an active pose using the **Pose 2.nma** pose file.
2. Set the **Start Frame** to Frame 39 and the **End Frame** to Frame 69.
3. Set the **Strength** property to 6.0, and the **Blend Period** property to 0.25. The blend period specifies the relative rate at which the character assumes the pose.

To add a third active pose:

Repeat the previous steps to add another active pose event to the Gymnast character timeline:

1. Load an active pose using the **Pose 3.nma** pose file.
2. Set the **Start Frame** to Frame 69 and the **End Frame** to Frame 113.
3. Set the **Strength** property to 10.0, and the **Blend Period** property to 0.4.

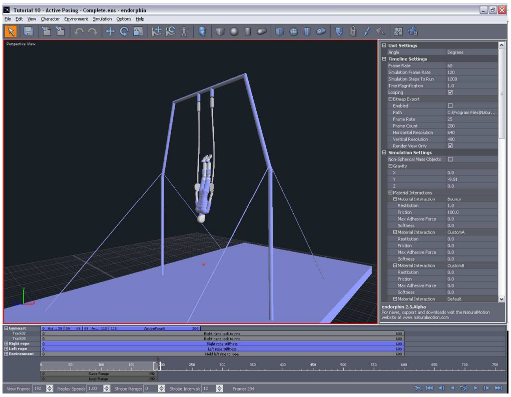
To add a fourth active pose:

Repeat the previous steps to add another active pose event to the Gymnast character timeline:

1. Load an active pose using the **Pose 4.nma** pose file.
2. Set the **Start Frame** to Frame 113 and the **End Frame** to Frame 264.
3. Set the **Strength** property to 10.0, and the **Blend Period** property to 0.4.
4. Simulate the **scene**. The effect of the four active poses is to move the gymnast's legs up, leading to a swing that reorients the character upside-down. The character maintains this pose until Frame 264.

   Keep in mind that **all** the character's motion has been generated by a combination of active poses, and the effect of those active poses on the character's dynamics. This is a good example of how a dynamic system like *endorphin* differs from a traditional keyframe animation system.

   Similarly, all the motion in the **ropes** is secondary motion caused by the motion of the gymnast. This realistic motion is an additional benefit of using a dynamics-based system.



Unit Settings	
Angle	Degrees
Timeline Settings	
Frame Rate	60
Simulation Frame Rate	120
Simulation Steps To Run	1200
Time Magnification	1.0
Looping	✓
Bitmap Export	
Enabled	☐
Path	C:\Program Files\Natur...
Frame Rate	25
Frame Count	200
Horizontal Resolution	640
Vertical Resolution	480
Render View Only	✓
Simulation Settings	
Non-Spherical Mass Objects	☐
Gravity	
X	0.0
Y	-9.81
Z	0.0
Material Interactions	
Material Interaction	Bouncy
Restitution	1.0
Friction	100.0
Max Adhesive Force	0.0
Softness	0.0
Material Interaction	CustomA
Restitution	0.0
Friction	0.0
Max Adhesive Force	0.0
Softness	0.0
Material Interaction	CustomB
Restitution	0.0
Friction	0.0
Max Adhesive Force	0.0
Softness	0.0
Material Interaction	Default

endorphin 2.5.Alpha
For news, support and downloads visit the NaturalMotion website at www.naturalmotion.com

Gymnast	0 Act... 39 · 39 69 · 69 Ac... 113 · 113 ActivePose8 264
Track02	0 Right hand lock to ring 600
Track03	0 Right hand lock to ring 600
Right rope	0 Right rope stiffness 600
Left rope	0 Left rope stiffness 600
Environment	0 Hold left ring to rope 600

0 Save Range 192
0 Loop Range 192

View Frame: 192 Replay Speed: 1.00 Strobe Range: 0 Strobe Interval: 12 Frame: 294

Further experiments

- Try changing the **Strength**, **Blend Period**, **Start Frame** and **End Frame** properties of the four active poses. Each combination will effect the overall motion of the gymnast.
- Try using the Pose Move and Pose Rotate tools to edit the individual active poses themselves.

Step 4: Completing the gymnastic movement

In this **optional** step, we will add four more **active poses** to the character timeline. These poses will continue drive the relative joint motion of the simulation character to complete the motion that was started in the previous steps.

To add a fifth active pose:

Repeat the previous steps to add another active pose event to the Gymnast character timeline:

1. Load an active pose using the **Pose 5.nma** pose file.
2. Set the **Start Frame** to Frame 264 and the **End Frame** to Frame 320.
3. Set the **Strength** property to 12.0, and the **Blend Period** property to 0.8.

To add a sixth active pose:

Repeat the previous steps to add another active pose event to the Gymnast character timeline:

1. Load an active pose using the **Pose 6.nma** pose file.
2. Set the **Start Frame** to Frame 320 and the **End Frame** to Frame 354.
3. Set the **Strength** property to 6.0, and the **Blend Period** property to 0.3.

To add a seventh active pose:

Repeat the previous steps to add another active pose event to the Gymnast character timeline:

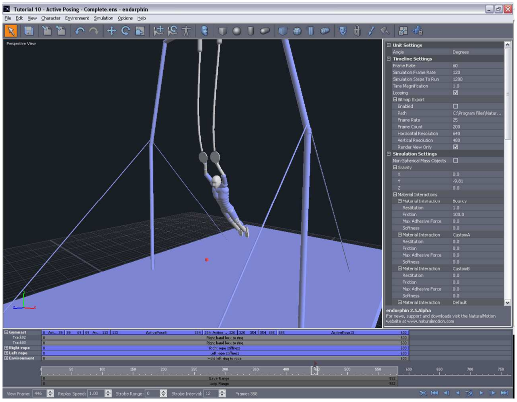
1. Load an active pose using the **Pose 7.nma** pose file.
2. Set the **Start Frame** to Frame 354 and the **End Frame** to Frame 385.
3. Set the **Strength** property to 5.0, and the **Blend Period** property to 0.4.

To add an eighth active pose:

Repeat the previous steps to add another active pose event to the Gymnast character timeline:

1. Load an active pose using the **Pose 8.nma** pose file.
2. Set the **Start Frame** to Frame 385 and the **End Frame** to Frame 600.
3. Set the **Strength** property to 5.0, and the **Blend Period** property to 0.4.

4. Simulate the **scene**. The effect of these four new active poses is to complete the overall gymnastic motion.


Tutorial 10 - Active Posing - Complete.ens - endorphin
File Edit View Character Environment Simulation Options Help
Perspective View
Gymnast
Right rope
Left rope
Environment

Unit Settings	
Angle	Degrees
Timeline Settings	
Frame Rate	60
Simulation Frame Rate	120
Simulation Steps To Run	1200
Time Magnification	1.0
Looping	✓
Bitmap Export	
Enabled	☐
Path	C:\Program Files\Natur…
Frame Rate	25
Frame Count	200
Horizontal Resolution	640
Vertical Resolution	480
Render View Only	✓
Simulation Settings	
Non-Spherical Mass Objects	☐
Gravity	
X	0.0
Y	-9.81
Z	0.0
Material Interactions	
Material Interaction	Bouncy
Restitution	1.0
Friction	100.0
Max Adhesive Force	0.0
Softness	0.0
Material Interaction	CustomA
Restitution	0.0
Friction	0.0
Max Adhesive Force	0.0
Softness	0.0
Material Interaction	CustomB
Restitution	0.0
Friction	0.0
Max Adhesive Force	0.0
Softness	0.0
Material Interaction	Default

endorphin 2.5.Alpha
For news, support and downloads visit the NaturalMotion website at www.naturalmotion.com

Further experiments

- Try changing the **Strength**, **Blend Period**, **Start Frame** and **End Frame** properties of the four active poses. Each combination will effect the overall motion of the gymnast. Also, try using the Pose Move and Pose Rotate tools to edit the individual active poses themselves.
- The **Left Rope** and **Right Rope** prop characters **also** have active poses applied to them. Unlike the active poses applied to the Gymnast character, the active poses on the prop characters are used to slightly **stiffen** these articulated objects. Try changing the **Strength** property of these stiffening active poses to make the ropes stiffer or more flexible.

Conclusion

You have used **active poses** to influence the motion of a simulated character, and experimented with different active pose strength and blend periods.

If you have any problems with this tutorial, open the corresponding scene **Resources\Tutorials\Tutorial 10 – Active Posing\Tutorial 10 – Active Posing - Complete.ens**. This scene is an example of how your scene should look if you have successfully followed all the steps in this tutorial.

Tutorial 11

Creating characters (advanced)

In this tutorial, you will explore the more advanced features of creating your own custom characters in *endorphin*. You should complete **Tutorial 05 - Creating characters** before working on this tutorial.

In particular, you will learn how to reshape your character to match an **external mesh** so that your character will interact with itself and the environment correctly.

Step 1: Creating new characters

In this step, we will create a simulation-reference character pair in *endorphin* using Character Edit Mode. The reference character will be based on the skeleton contained in an FBX file.

This step is similar to Step 1 of Tutorial 5. See that step for more detail.

To create a new simulation-reference character pair:

1. Launch *endorphin*.
2. Select **File > New Character…**. This displays the **New Character** dialog.
3. In the **Create Character Using** group, ensure that **Standard Simulation Character** is selected.
4. Turn on the **Create Reference Character Using** setting, and click the Browse button to select the file **Resources\Tutorials\Tutorial 11 – Creating Characters (Advanced)\MyCharacter.fbx**. Click **OK**.
5. In the **Import Options** dialog appears, select the **root** node from the node hierarchy, and click **OK**.
6. The new character pair is displayed in Character Edit Mode.

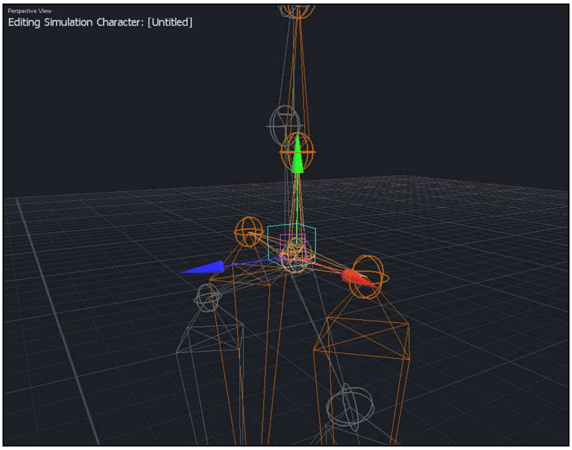
Step 2: Adjusting the root position

In this step, we will be **reshaping** the simulation character to match the corresponding reference character. This involves using the Move tool to snap simulation character joints to reference character joints. The first step is to ensure that the **root joint** position of the simulation character exactly coincides with the root joint position of its reference.

This step is similar to Step 2 of Tutorial 5. See that step for more detail.

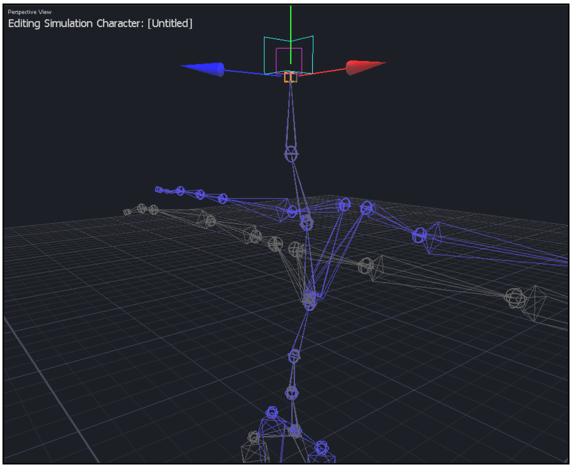
To match the root position of the simulation and reference characters:

1. Right-click in the viewport and select **Skeletal View**.
2. Zoom in on the hips of the two characters.
3. Activate the **Move** tool and select the **root** joint of the simulation character.
4. Snap the root joint of the simulation character with the corresponding root joint of the reference character. When you are snapping joints using the Move tool, ensure that you are using the **purple square** manipulator of the Move tool. If you use the axial or planar manipulators of the Move tool, joint snapping will not occur.



5. Select the **LowerSpineJoint** of the simulation character, and snap it to the **Spine1** joint of the reference character. Working up the spine, snap all the other joints into place, all the way up to the head.

   When you snap joints together, they become **locked** together. When a joint is locked, its position is not affected when you move or rotate other joints. For example, if you lock the shoulder and wrist joints, you can move and rotate the elbow joint without disrupting the snapped positions of the elbow or wrist. Of course, you can always use the Move tool to directly move the elbow and wrist joints, which then **unlocks** these joints.



Step 3: Reshaping the joint hierarchy

In this step, we will continue the process of **reshaping** the simulation character to match its corresponding reference. Starting at the simulation character root joint, and moving outwards in the direction of its extremities, we need to adjust the length and orientation of each joint in the simulation character to match the corresponding joint of the reference character.

This step is similar to Step 3 of **Tutorial 5**. See that step for more detail.

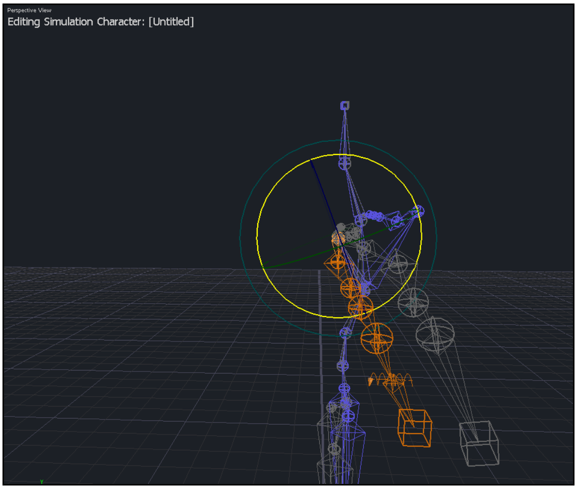
To reshape the joints of the simulation character:

1. Select **LeftClavicleJoint** of the simulation character and snap it to the corresponding joint of the reference character.

2. If you inspect the **LeftFingersJoint** of the simulation character, you will notice a helix that indicates this is a **Hinge** Joint. Most other simulation character joints are **Skeletal** (ball-and-socket) joints.

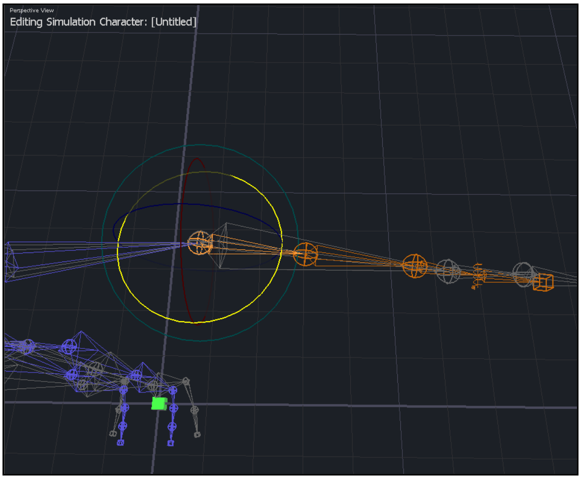
   The joint represents the simulation character's **knuckles**. Ideally, this joint should be aligned so that it is **horizontal** when the arms are outstretched in a T-pose. With the **LeftClavicleJoint** still selected, rotate it about the **X axis** using the **red circle** manipulator. Rotate it such that the knuckles are

horizontal again. This is a **clockwise** rotation, when looking down the left arm from the head to the hand.



3. With the **LeftClavicleJoint** in place, snap the **LeftShoulderJoint** and **LeftElbowJoint** into place.

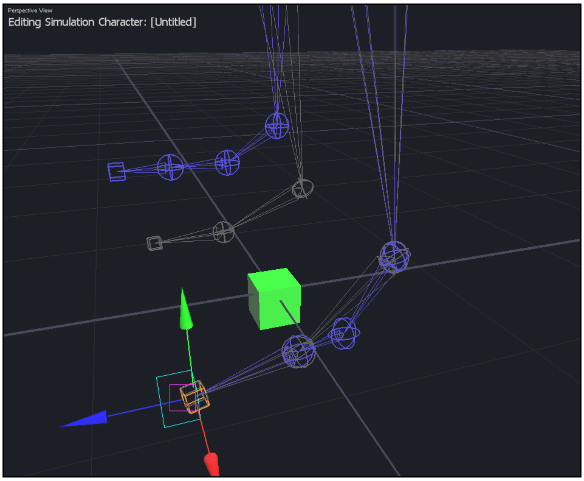
4. With the **LeftElbowJoint** still selected, rotate it about the **Y axis** so that the **LeftForeArmTwist** joint is aligned with the forearm of the reference character.



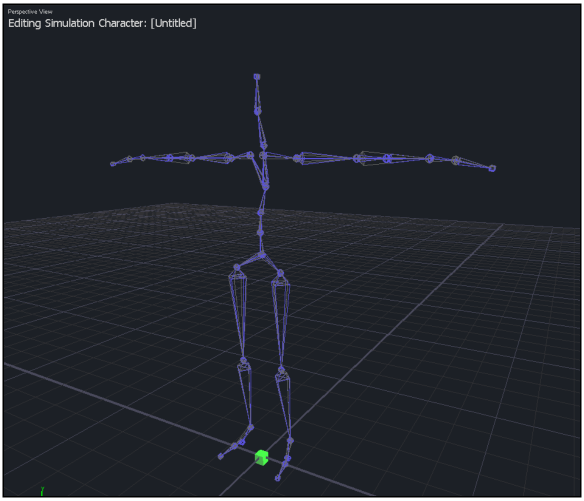
5. Bypass the **LeftForeArmTwist** joint, and snap the wrist and finger joints into place.

6. Click the **Mirror Left to Right** button in the Main Toolbar. This mirrors the changes you have made from the left arm to the right arm.

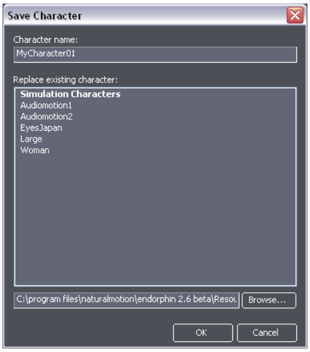
7. Select the **LeftHipJoint** of the simulation character, and snap it into place.
8. Continue working down the joints of the left leg until you reach the **LeftAnkleJoint**. Snap this joint to the ankle of the reference skeleton.
9. Select the **LeftMidFootJoint** and move it until it is roughly halfway along the foot joint in the **XZ plane** and vertically above the toe joint in the **Y axis**.
10. Snap the **LeftToesJoint** and the **LeftToesEndJoint** into place.



11. Click the **Mirror Left to Right** button in the Main Toolbar. This mirrors the changes you have made from the left leg to the right leg.



12. Select **File > Save Character**. Save the character pair as **MyCharacter01**. The viewport label will update to display this new character name.



MyCharacter01

Simulation Characters
Audiomotion1
Audiomotion2
EyesJapan
Large
Woman

C:\program files\naturalmotion\endorphin 2.6 beta\Resou

Step 4: Displaying joint limits

In this step, we will display some of the **joint limits** of the simulation character.

Each joint in an *endorphin* character has a corresponding joint limit, which defines its allowable range of movement. The default joint limits of the Standard Simulation Character are set for a standard adult human, and often will not need to be adjusted. To avoid cluttering the viewports, all joint limits are hidden by default. You can show and hide joint limits by using the Node View.

After you have reshaped a simulation character to match its reference character, it is good practice to review the joint limits to ensure that all the reshaped joints are correctly aligned. In the following example, we will examine the joint limits of the shoulders.

To display the shoulder joint limits:

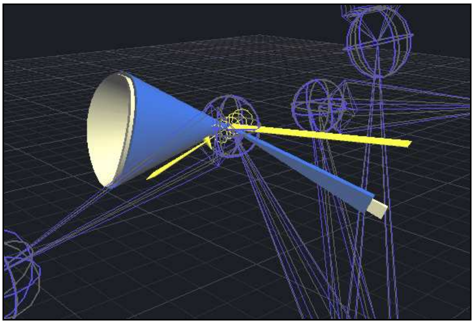
1. Ensure that the simulation character is active. If the reference character is active, click the **Activate Simulation Character** button in the Main Toolbar. The viewport label should display **Simulation Character - MyCharacter01**.

2. Ensure that the **Node View** is visible. If it is not visible, select **View > Node View** to display the Node View. The keyboard shortcut is **N**.

3. Select the **LeftClavicleJoint**.

4. In the Node View, click the joint limit icons next to the **LeftClavicleJointLimit** and **LeftClavicleJointLimitOffset** nodes. This displays the corresponding joint limit graphics in the viewport. The

corresponding icons in the viewport will be drawn solidly—rather than partially transparently—when the joint limits are shown in the viewport.



Swing and twist limits

Skeletal joint limits are displayed using separate **swing** and **twist** limit graphics. Hinge joints are displayed with twist limit graphics only.

The **swing** limit is displayed as a blue cone that runs along the direction of the bone, and indicates the allowable swing range of the joint in the two directions perpendicular to the bone. The **twist** limit is a blue arc that sits in a plane perpendicular to the bone, and indicates the allowable twist range of the joint.

Swing limits and twist limits each have a **hard** limit and a **soft** limit. The hard limit is drawn in blue, and the soft limit is drawn in white. Joints can move freely inside their soft limits, and cannot move at all past their hard limits. Joints can move between the soft and hard limits, but with increasing resistance.

Each limit also has a yellow **offset** indicator. A joint limit offset indicates the current position of the joint with respect to its joint limits.

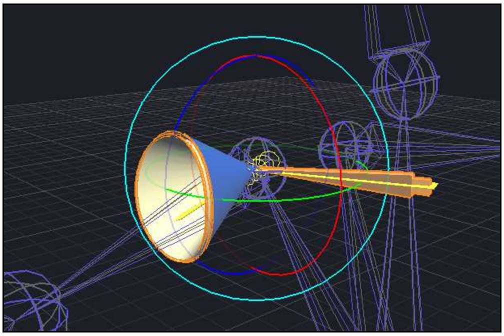
Note When you are editing characters in Character Edit Mode, it is possible to reorient joint limits so that their offsets lie **outside** their valid joint limit range. It is important to ensure this is corrected **before** using the character in a simulation. Otherwise, you will notice the character **popping** during a simulation—its joints will rapidly rotate back into their allowable ranges during the first few frames of simulation.

Step 5: Adjusting the shoulder joint limits

In this step, we will adjust the **shoulder joint limits** of the simulation character.

To adjust the shoulder joint limits:

1. Display the **LeftClavicleJoint** limit (see Step 4 for more detail).
2. Experiment with the joint limit by adjusting some of the swing and twist limit values and orientations using the Property Editor. Each time you change a joint limit value in the Property Editor, you will see an update in the corresponding joint limit graphic in the viewport. Do **not** save these changes.
3. Discard these changes by **reopening** the character file without saving the changes.
4. Select the **LeftClavicleJoint** limit in the viewport. It will be displayed using the orange highlight colour.
5. Select the **Rotate** tool.
6. Toggle the tool coordinate system so that you are rotating in **local** coordinates. To do so, select **Edit > Toggle Coordinate System**. The keyboard shortcut is **X**.
7. Rotate the joint limit so that the joint limit swing offset is located **within** the joint swing limit. Ideally, the swing limit cone should be relatively **high** in relation to the offset. That is, we want the shoulders to be able to move up **more** than they can move down—which reflects typical human anatomy. The precise location of the offset relative to the limit depends on the freedom of movement you want to specify for the character.



8. Select **Character > Mirror Left To Right**, or click the **Mirror Left To Right** button in the Main Toolbar, to mirror the changes in the left shoulder joint limit across to the right shoulder joint limit.

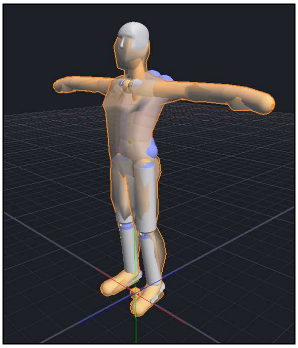
9. In the Node View, hide the joint limit graphics. The easiest way to do this is to right-click on the character node and select **Hide Type > Joint Limits**.
10. Select **File > Save Character**, or click the **Save** button in the Main Toolbar, to save the character. The keyboard shortcut is **Ctrl+S**.

Step 6: Importing a skin mesh

In this step, we will import an **OBJ** file that represents the surface mesh—also called the skin mesh—of your reference character. This mesh will be contained as a helper **graphical object**, which is displayed in the viewport. Helper graphical objects are very useful when you need to accurately reshape the mass and collision objects of a simulation character to match the shape and size of your own character.

To import a helper skin mesh:

1. Right-click in the viewport and select **Shaded View**. This will display both the simulation and reference characters in shaded, rather than skeletal, mode.
2. Select the reference character by clicking the **Activate Reference Character** button in the Main Toolbar, or by right-clicking in the viewport and selecting **Select Reference Character**.
3. Select **File > Import** or click the Import button in the Main Toolbar to import a file. The keyboard shortcut is **I**.
4. Browse to select the file **MyCharacter.obj**.
5. In the **Import Options** dialog, leave all the settings as default and click **OK**. A new graphical object is created belonging to the reference character.



6. In the **Node View**, right-click the reference character and select **Hide All**. Then, right-click on the **MyCharacter** graphical object in the Node View and select **Show**. The reference character will be hidden, except for its graphical object representing its skin mesh.

7. **Activate** the simulation character. This will deselect the graphical object. The graphical object will no longer be selectable in the viewport, because it belongs to the inactive reference character. However, you can still select the graphical object using the Node View.

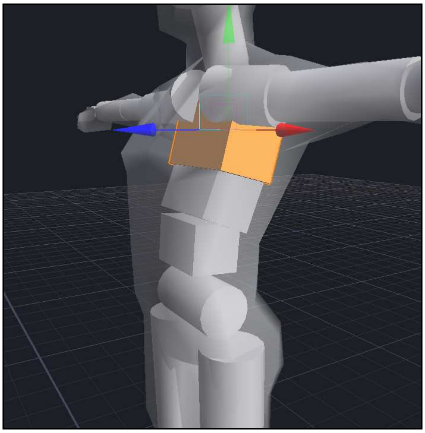
Step 7: Reshaping the mass objects

In this step, we will adjust the **mass objects** of the simulation character by changing their size, position and orientation. This is often called **reshaping** the mass objects.

Each joint in a character (other than the end joints) has a corresponding mass object. You cannot add or remove mass objects, but you can reposition them relative to the joint. You can also resize them and change their density. The goal is to obtain a mass distribution in the simulation character that best reflects the real mass distribution of your own character. Keep in mind that mass objects are not involved with collisions.

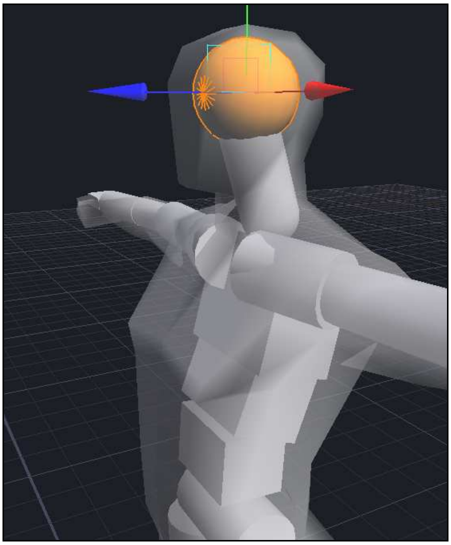
To reshape the mass objects:

1. Right-click in the viewport and turn off **Collision Objects**. This will hide all collision objects in the viewport. Also, right-click in the viewport and select **Joints** and **Bones**. This will display joints, which is useful when positioning mass objects.
2. Select the **PelvisMass** mass object.
3. Use the **Move** tool to move this mass object to a more central position inside the hip region of the skin mesh. Keep in mind that when you are positioning a mass object, you are positioning it with respect to its parent joint.
4. Use the **Scale** tool to adjust the size of this mass object so that it fits the hip region of the skin mesh. Keep in mind that mass objects are often heavier than you might think. Mass objects are completely solid objects, and their overall mass is defined by their size, shape and density. In general, you should not make mass objects too big. The Standard Simulation Character has been designed so that the mass distribution reflects the typical adult male human.
5. Continue reshaping mass objects along the spine joints. The **UpperSpineMass** mass object will also need rotating.



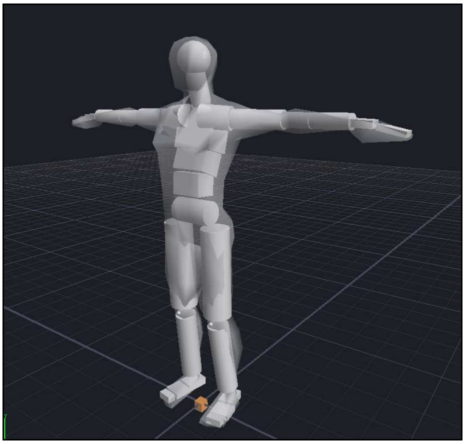
6. Continue reshaping mass objects throughout the left-hand side of the character. When you are reshaping the **HeadMass** mass object, you will need to hide the **HeadGraphic** graphical object in order to view the mass object that is hidden inside it. Right-click on the **HeadGraphic** graphical object in the Node View and select **Hide**.

7. Select the spherical **HeadMass** mass object and use the Move tool to position it correctly at the centre of the skin mesh head region.



8. Continue reshaping the mass objects in the left arm and left leg. When you are reshaping mass objects in the hands and feet, do not make them too small. Very small mass objects may be ignored by the simulation.

9. Finally, select **Character > Mirror Left To Right**, or click the **Mirror Left To Right** button in the Main Toolbar to mirror the reshaped mass objects to the right-hand side of the character.



Step 8: Reshaping the collision objects

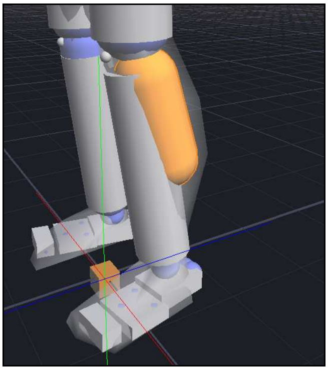
In this step, we will adjust the collision **objects** of the simulation character by changing their size, position and orientation. This is often called **reshaping** the collision objects.

Each mass object in a character has zero or more corresponding collision objects. You can add and remove collision objects, and you can also reposition them relative to their parent mass objects. You can also resize them and change their material types. The goal is to obtain a collision surface in the simulation character that best reflects the real skin mesh shape of your own character.

To reshape the collision objects:

1. Right-click in the viewport and turn on **Collision Objects**. Also, turn off **Joints** and **Bones**.
2. Zoom in to view the lower left leg.
3. Select the **LeftLowerLegCollision** object and use the Move and Scale tools to reposition and resize this object so that it matches the shape of calf region of the mesh. The collision object should not protrude outside the mesh, but fit as closely inside as possible.

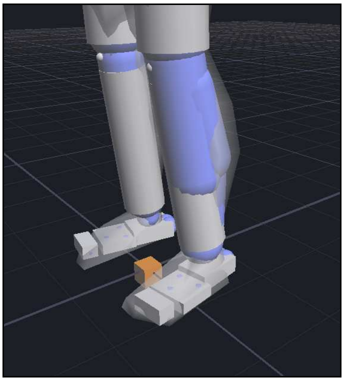
   Note We are modeling a complex surface using relatively simple volumes such as sphyls, so there will not be a complete match between the mesh surface and your own collision surfaces. Using more collision objects will result in a more accurate representation of the skin mesh, but at the expense of slower simulation times.

4. You can also create additional collision objects. Select the **LeftLowerLegMass** object and select **Character > Create Sphyl Collision Object**, or click the **Create Sphyl Collision Object** button in the Main Toolbar. This will create a new sphyl collision object as a child of

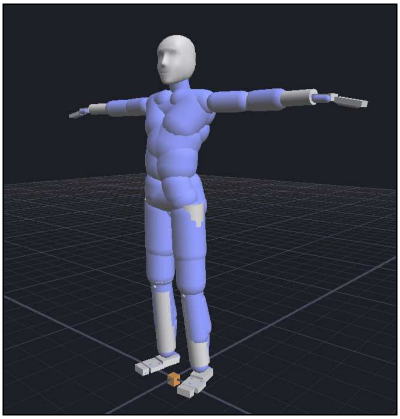
the selected mass object. Adjust the size, position and orientation of this new collision object so that it fills the lower knee area of the left leg.

5. By default, new collision objects will have their **Collides With Parent**, **Collides With Siblings** and **Collides With Ancestors** settings all turned **on**. You will need to turn these settings **off** where appropriate, in order to allow for any required movement. To modify these settings, select the collision object and use the Property Editor to make these changes.



6. Continue reshaping collision objects throughout the left-hand side if the character.

7. Finally, select **Character > Mirror Left To Right**, or click the **Mirror Left To Right** button in the Main Toolbar to mirror the reshaped collision objects to the right-hand side of the character. This command will also create or delete collision objects on the right-hand side of the character, to match the collision objects on the left-hand side of the character.



Conclusion

In this tutorial, you have learned more about using Character Edit Mode to create characters. You have learned how to display and edit joint limits, and how to use imported mesh objects as guides when reshaping mass objects and collision objects.

Tutorial 12

Using motion transfer

In this tutorial, you will learn more about using **motion transfer** to map animation data from your original reference character onto a reshaped simulation character.

The most common use of motion transfer is during data **import** and **export**. The Import Options and Export Options dialogs allow you to specify a reference character for use as an intermediary when you are importing or exporting animated data.

You can also use motion transfer within a simulation itself, in the form of **Motion Transfer events**. These allow you to drive the motion of a simulation character using a combination of animation data and *endorphin* events such as behaviours and active poses.

This tutorial covers the motion transfer events, and introduces you to the concept of **rig nodes**. Rig nodes allow you to specify the degree to which motion is transferred for each mass object.

Step 1: Adding a character pair to a scene

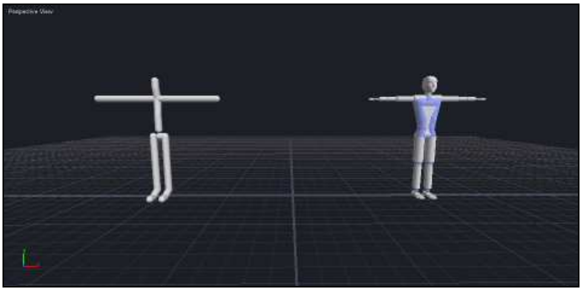
In this step, we will create a new scene, and add a simulation-reference character pair to the scene.

To add a simulation-reference character pair to a scene:

1. Launch *endorphin* and delete the default character.
2. Select **Character > Add Character…** or click the **Add Character** button in the Main Toolbar to display the Add Character dialog.
3. Click the Browse… button and select the folder **Resources\Tutorials\Tutorial 12 - Using Motion Transfer**. Click **OK**. This updates the working folder for characters.
4. The character list will be updated to reflect the characters found in the working folder. Select the **AudioMotion1** character from the list. Turn on the **Include Reference Character** setting and click **OK**. Two characters

will be added to the scene—the AudioMotion1 simulation character, and the AudioMotion1 reference character. The simulation character is the same shape and size as the reference character.

5. New characters are added at the origin of the scene, which means the AudioMotion1 simulation and reference characters are now superimposed. Select the **AudioMotion1_ref** reference character and move it 2 metres along the X axis. Select the **AudioMotion1** simulation character and move it 2 metres along the X axis in the other direction.



Step 2: Importing animation onto the reference character

In this step, we will import animation data directly onto the reference character in a scene. You do **not** need to turn on the **Transfer From Reference Character** setting. This is because we are importing animation data directly onto its corresponding reference character. You only need to use Auto Motion Transfer when you are importing animation data onto simulation characters.

To import animation data onto the reference character:

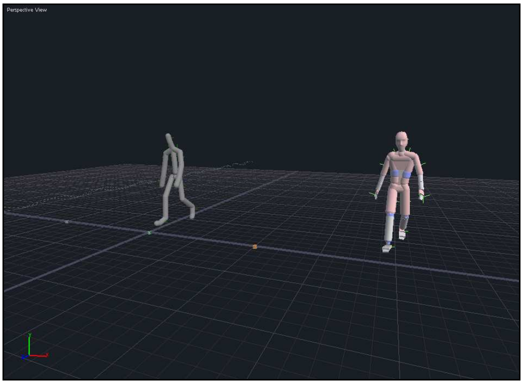
1. Select the **AudioMotion1_ref** reference character.
2. Select **File > Import…**, or click the **Import** button on the Main Toolbar. The keyboard shortcut is **I**. Select the file **Resources\Tutorials\Tutorial 12 - Using Motion Transfer\AM_walk.fbx**.
3. Select the **Hips** node from the node hierarchy.
4. Turn on the **Include Parent Transforms** setting.
5. Click **OK**. The animation data is imported as a new animation event on the **AudioMotion1_ref** character timeline.

Step 3: Transferring motion to the simulation character

In this step, we will add a Motion Transfer event to the simulation character timeline. This will allow us to transfer the motion imported in Step 2 onto the simulation character.

To transfer motion to the simulation character:

1. Right-click on the **AudioMotion1** character timeline in the Timeline Editor, and select **Create Motion Transfer Event**.
2. Adjust the Start Frame so that is starts one frame **after** the animation event. When an animation event is applied to a character, the first frame it affects is the **next** frame along. For example, if an animation event is applied at frame 200, you will first notice its effect at frame 201. If you would like a motion transfer event to be applied to the results ofan animation event, it needs to start at least one frame after the animation event begins.
3. Adjust the End Frame to match the end frame of the animation event.
4. Select the new motion transfer event.
5. In the Property Editor, locate the **[Select]** command hotlink adjacent to the **Source Character** property. Click this hotlink to enter selection mode, and select the **Audiomotion1_ref** reference character in the viewport, Node View or Timeline Editor. **Right-click** in the viewport to leave this selection mode. The AudioMotion1_ref reference character has now been identified as the motion transfer source.
6. Simulate the scene. The **AudioMotion1_ref** reference character is driven entirely by the animation data contained in its animation event. The **AudioMotion1** simulation character is dynamically driven by the reference character via the motion transfer event. Keep in mind that the AudioMotion1 simulation character is **simulated** during the motion transfer process. This means you can modify the motion of the simulation character with other *endorphin* functionality such as behaviours and active poses.



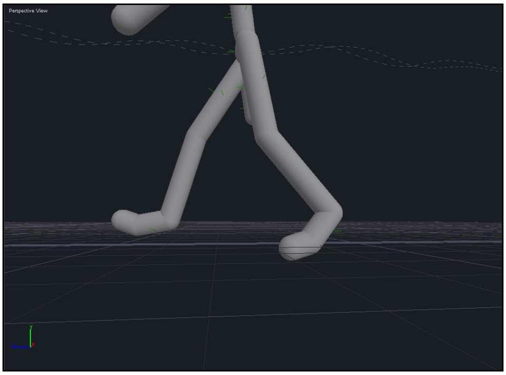
Step 4: Controlling motion transfer with active poses

In this step, we will use **active poses** to help control the transferred motion on the target character.

Motion transfer involves driving **some** of the mass objects in a target character using the motion of corresponding mass objects in a source character. The mass objects that are **not** directly driven are able to move freely, subject to conditions such as joint limit constraints. Occasionally this can lead to mass objects wobbling in an undesirable manner. Adding a soft active pose is a useful technique to help smooth out unwanted artifacts of the motion transfer process.

To control motion transfer with an active pose:

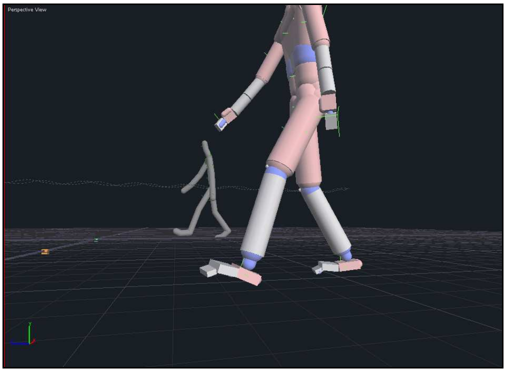
1. Zoom into the feet of the **reference** character.
2. If you examine the feet of the reference character, you will notice that it has **two-bone** feet. The toes of the reference character pass through the ground plane during the walk cycle. You can use **camera tracking** to follow the motion of the reference character feet.



3. Zoom into the feet of the **simulation** character.

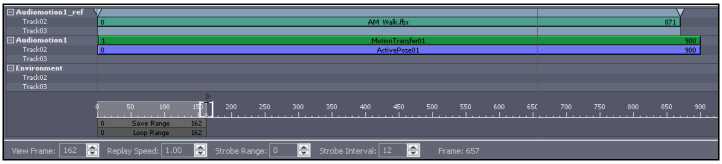
4. If you examine the feet of the simulation character, you will notice that it has **three-bone** feet. The extra bones are designed to improve **foot-roll** motion. The toes of the simulation character do **not** pass through the ground plane during the walk cycle. This clean-up happens automatically, due to the interaction of the collision objects in the feet of the simulation character.

5. However, this additional joint has a corresponding mass object which has no corresponding rig node connection. As a result, this joint rotates freely during the walk cycle. We are going to add an active pose to the simulation character to prevent this unwanted toe rotation.



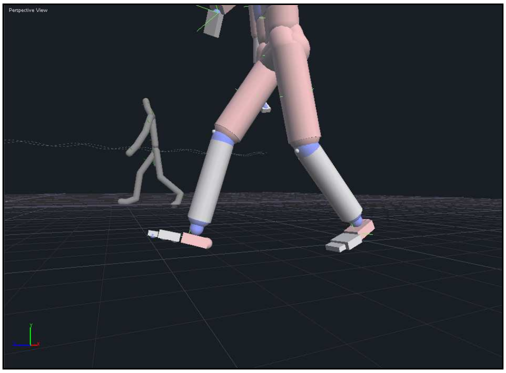
6. Right-click on the AudioMotion1 simulation character timeline in the Timeline Editor, and select **Create Active Pose Event**. Ensure that the active pose has its Start Frame set to Frame 0, and that its duration matches the duration of the motion transfer event.

   The active pose adds some additional joint stiffness to all the joints in the simulation character. This joint stiffness does not affect joints driven by the motion transfer event. However, it will prevent free rotation on joints that are **not** driven by motion transfer.



View Frame: 162 Replay Speed: 1.00 Strobe Range: 0 Strobe Interval: 12 Frame: 657

7. Simulate the scene again. You should find that the foot now behaves correctly, with no foot-ground slip, foot-ground penetration or toe wobble.



Step 5: Working with motion transfer rig groups

In this step we will create new motion transfer **rig groups**.

Introducing rig groups and rig nodes

Rig groups are composed of **rig nodes**. Each rig node has corresponding positional and angular **strengths** that specify the degree to which the rig node can transfer motion from a source mass object to a target mass object.

Until now, we have been using the **Standard Motion Transfer** rig group. You will use this rig group for transferring **all** the motion from a source character to a target character. It can be useful to create additional custom rig groups when you want to only partially transfer motion. For example, you might want to use motion transfer to drive a character's torso and legs, and use a behaviour to drive the character's arms and head.

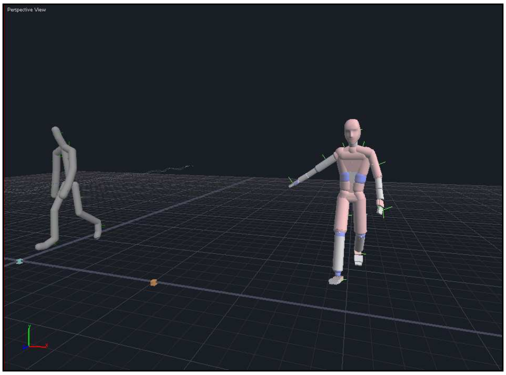
To create a new rig group:

1. Display the Motion Transfer Editor by selecting **View > Motion Transfer Editor**. The keyboard shortcut is **M**. The Motion Transfer Editor is automatically displayed when you enter Character Edit Mode.
2. Select the **AudioMotion1** simulation character. The Motion Transfer Editor displays the settings for this character.
3. Browse to the **Strengths** tab. This page displays a list of available rig groups. You should find that the only rig group is the Standard Motion Transfer rig group.
4. Click the **Add…** button to create a new rig group. Name the new rig group **Right Arm Free** and click **OK**.
5. Select the **RightUpperArm** and **RightHand** rig nodes. You can hold down the **Ctrl** key to select multiple rig nodes.
6. Change the rig node mode of the selected rig nodes to **NoHold**. This frees the right arm so that it will not be driven by motion transfer.

To use the new rig group with motion transfer:

1. Select the motion transfer event on the AudioMotion1 simulation character timeline.
2. Use the Property Editor to change the **Rig Group** property of the motion transfer event to **Right Arm Free**. When the scene is simulated, the motion transfer will now use the new rig group that you have created, and the right arm of the simulation character will not be driven by the walk cycle.
3. Simulate the scene. You will notice that the mass objects making up the right arm of the simulation character are no longer displayed in red. Only mass objects that are driven by rig nodes are displayed in red during a motion transfer event.

   Also, notice that the right arm of the simulation character attempts to reach out to its side. This motion is due to the presence of the weak **active pose** that was added in Step 4. That active pose was a T-pose, in which the right arm was placed out to the side of the character. However, we can change the pose used by this active pose event.

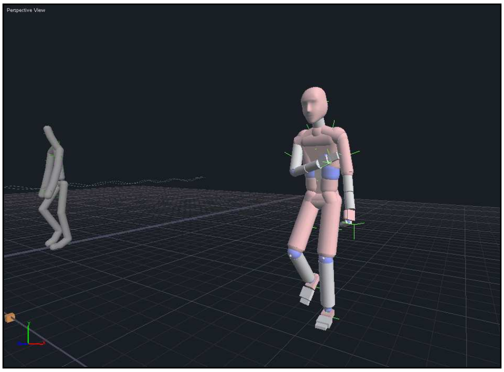


4. Select the active pose event in the **AudioMotion1** simulation character timeline.

5. Right-click on the active pose event and select **Load Pose...**. Browse to select the **Holding Shield.nma** pose file. In this pose, the character has its right arm across its chest, as though it were carrying a shield-like item.

6. Deselect the active pose event.

7. Simulate the scene again. The character will now lift its arm towards its chest and hold it there during the walk cycle. Meanwhile, the legs, torso and left arm of the simulation character continue to be driven by the motion transfer event.



8. Experiment with this scene. For example, try changing the **strength** of the active pose event. This will change the degree to which the active pose affects the motion of the right arm.

   Also, try adding a mass object to the scene and attaching it to the right hand of the simulation character with a **JoinBodies** constraint. This mass object represents the mass of the shield carried by the simulation character. Note how the position of the arm is affected by the mass of this mass object.

Conclusion

You have used motion transfer events to transfer motion from an animation event onto a simulation character. You have also created additional motion transfer rig groups to modify the way motion is transferred. Finally, you have mixed motion transfer events with active pose events to control the motion of joints that are not driven by the motion transfer.

If you have any problems with this tutorial, open the corresponding scene **Resources\Tutorials\Tutorial 12 – Using Motion Transfer\Tutorial 12 – Using Motion Transfer - Complete.ens**. This scene is an example of how your scene should look if you have successfully followed all the steps in this tutorial.

Tutorial 13

Advanced motion transfer and prop interaction

In this tutorial, you will **reuse** existing animation data in new ways by using motion transfer events in conjunction with other *endorphin* features, such as mass objects, constraints and active poses. The ability to rapidly reuse motion-captured or keyframed animation data is an important aspect of *endorphin*.

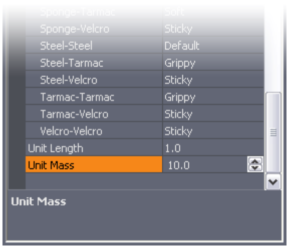
In particular, you will reuse a standard walk cycle animation and modify it so that the character appears to hold a weapon while walking. You can then adjust the mass of the weapon and the strength of the character's arms to generate different effects.

Step 1: Adding a character pair to a scene

In this step, we will create a new scene, and add a simulation-reference character pair to the scene.

To add a simulation-reference character pair to a scene:

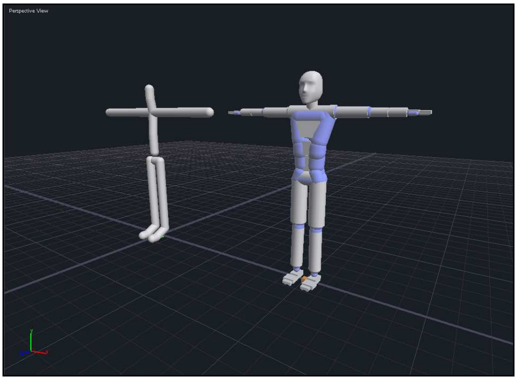
1. Launch *endorphin* and delete the default character.
2. Locate the **Unit Mass** property in Property Editor, and set its new value to 10.0. This property reflects the characteristic, or typical, mass in the scene. Increasing this value is useful when you want to add larger masses to the scene.



Sponge-Tarmac	Soft
Sponge-Velcro	Sticky
Steel-Steel	Default
Steel-Tarmac	Grippy
Steel-Velcro	Sticky
Tarmac-Tarmac	Grippy
Tarmac-Velcro	Sticky
Velcro-Velcro	Sticky
Unit Length	1.0
Unit Mass	10.0

Unit Mass

3. Select **Character > Add Character…** or click the **Add Character** button in the Main Toolbar to display the Add Character dialog.
4. Click the Browse… button and select the folder **Resources\Tutorials\Tutorial 13 - Advanced Motion Transfer And Prop Interaction**. Click **OK**. This updates the working folder for characters.
5. The character list will be updated to reflect the characters found in the working folder. Select the **AudioMotion2** character from the list. Turn on the **Include Reference Character** setting and click **OK**. Two characters will be added to the scene—the AudioMotion2 simulation character, and the AudioMotion2 reference character. The simulation character is the same shape and size as the reference character.
6. New characters are added at the origin of the scene, which means the AudioMotion2 simulation and reference characters are now superimposed. Select the **AudioMotion2** simulation character and move it 2 metres along the X axis away from the origin.

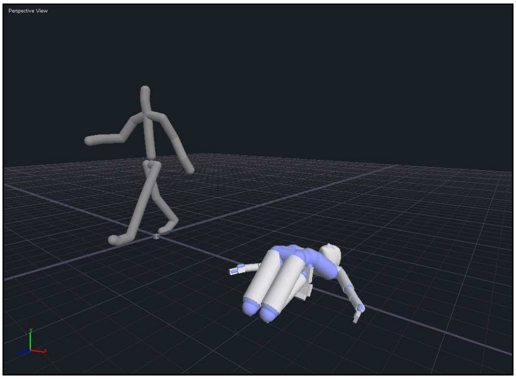


Step 2: Importing animation onto the reference character

In this step, we will import animation data directly onto the reference character in a scene. You do **not** need to turn on the **Transfer From Reference Character** setting. This is because we are importing animation data directly onto its corresponding reference character. You only need to use Auto Motion Transfer when you are importing animation data onto simulation characters.

To import animation data onto the reference character:

1. Select the **AudioMotion2_ref** reference character.
2. Select **File > Import…**, or click the **Import** button on the Main Toolbar. The keyboard shortcut is **I**. Select the file **Resources\Tutorials\Tutorial 13 - Advanced Motion Transfer And Prop Interaction\Walk.fbx**.
3. Select the **Hips** node from the node hierarchy.
4. Turn on the **Include Parent Transforms** setting.
5. Click **OK**. The animation data is imported as a new animation event on the **AudioMotion2_ref** character timeline.
6. Simulate the scene. The reference character is driven by the walk animation. The simulation character collapses, because we are not yet transferring the walk cycle across to the simulation character.

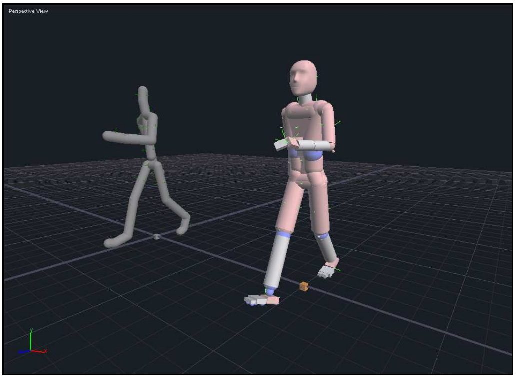


Step 3: Transferring motion to the simulation character

In this step, we will add a Motion Transfer event to the simulation character timeline. This will allow us to transfer the motion imported in Step 2 onto the simulation character.

To transfer motion to the simulation character:

1. Right-click on the **AudioMotion2** character timeline in the Timeline Editor, and select **Create Motion Transfer Event**. Adjust the Start Frame so that is starts one frame after the animation event. Adjust the End Frame to match the end frame of the animation event.
2. Select the new motion transfer event.
3. In the Property Editor, locate the **[Select]** command hotlink adjacent to the **Source Character** property. Click this hotlink to enter selection mode, and select the **Audiomotion2_ref** reference character in the viewport, Node View or Timeline Editor. **Right-click** in the viewport to leave this selection mode. The AudioMotion1_ref reference character has now been identified as the motion transfer source.
4. Right-click on the AudioMotion2 simulation character timeline in the Timeline Editor, and select **Create Active Pose Event**. Ensure that the active pose has its Start Frame set to Frame 0, and that its duration matches the duration of the motion transfer event. This active pose helps smooth out the motion of any joints that are not driven by the motion transfer event.



Step 4: Preparing the initial pose

In this step, we will adjust the start pose of the simulation character. Instead of using a T-pose, it will use the pose from the first frame of the animation. This will avoid the sudden jump that you see between Frame 0 and Frame 1.

To set the initial pose of the simulation character:

1. **Simulate** for a short duration, so that at least a few frames of animation have been stored in the frame buffer, and then stop the simulation.
2. Move the Time Slider to Frame 2. The AudioMotion2 simulation character should be in the first frame of the animation.
3. Select the AudioMotion2 character, and then select **Character > Save Pose…**.
4. Save this pose as **Start.nma**. This creates an *endorphin* **.nma** pose file containing the character pose.
5. Move the Time Slider to Frame 0.
6. Ensure that the AudioMotion2 simulation character is still selected.
7. Select **Character > Load Pose…**, and browse for the **Start.nma** pose file. Do not modify any of the Load Pose settings, and click **OK**.
8. Simulate the scene. The AudioMotion2 simulation character motion should now be smooth, with no discontinuities between Frame 0 and Frame 1.

Step 5: Adding a weapon as a prop character

In this step, we will add a new **prop** character to the scene. Prop characters are often used to model weapons, vehicles and other objects that can interact with simulation characters. In this step, the prop character will represent a rifle.

To add a weapon as a prop character:

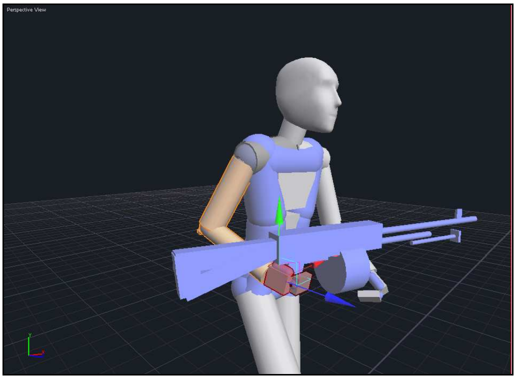
1. Move the Time Slider to Frame 0.
2. Select **Character > Add Character…**, or click the **Add Character** button in the Main Toolbar, to display the Add Character dialog.
3. Select the **Gun Large** prop character from the **Prop Characters** list.

4. Use the Move and Rotate tools to reposition the Gun Large character so that its grip is close to the left hand of the AudioMotion2 simulation character.

5. Activate the Pose Move tool.

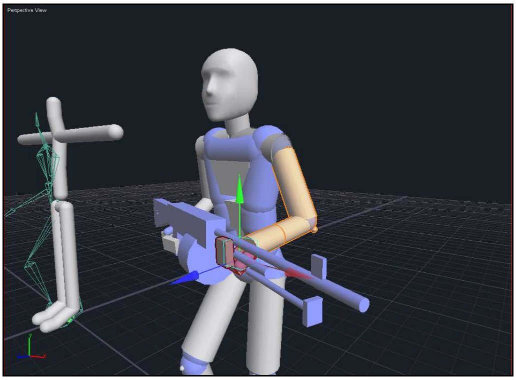
6. Right-click on the **RightShoulderMass** mass object to **lock** it.

7. Select the **RightHandMass** mass object, and use the Pose Move and Pose Rotate tools to pose the hand so that it is holding the gun.



8. Unlock the **RightShoulderMass**, and lock the **LeftShoulderMass**.

9. Select the **LeftHandMass** mass object, and use the Pose Move and Pose Rotate tools to pose the hand so that it is holding the gun.



Step 6: Controlling motion transfer with additional active poses

In this step, we will use an **active pose** to help control the transferred motion on the target character.

To control motion transfer with additional active poses:

1. Right-click on the **AudioMotion2** simulation character timeline and select **Create Active Pose Event**.
2. Adjust the Start Frame of the new active pose event so that it starts at Frame 0. Adjust the End Frame of the new active pose event so that it matches the end frame of the animation event on the AudioMotion2_ref character timeline.
3. Select the new active pose event.
4. In the Property Editor, set its **Strength** property to 3.0. This is an increase on the default active pose strength of 1.0.
5. Locate the **[Select]** command hotlink adjacent to the **Target Joints** property, and click it. This places *endorphin* in selection mode, and allows you to specify the joints affected by the active pose. By default, new active poses affect every joint in the character, unless you explicitly modify the Target Joints property.
6. In the viewport, box-select the **top half** of the AudioMotion2 simulation character.
7. **Right-click** in the viewport to accept the new set of active pose target joints.
8. Select the active pose event created in Step 3.
9. Modify its Target Joints property by box-selecting the **bottom half** of the AudioMotion2 simulation character.
10. By default, the two active poses will be called **ActivePose01** and **ActivePose02**. You can rename these events by editing their **Name** properties in the Property Editor. For example, you might rename these events ActivePoseLowerBody and ActivePoseUpperBody.

    Renaming timeline events can make it easier to quickly **read** a timeline. This is particularly for constraint and active pose events, since these events display their names on their timeline markers.

Step 7: Working with motion transfer rig groups

In this step we will create a new motion transfer **rig group**, so that the new active pose can control the upper body of the AudioMotion2 simulation character.

To create a new rig group:

1. Display the Motion Transfer Editor by selecting **View > Motion Transfer Editor**. The keyboard shortcut is **M**. The Motion Transfer Editor is automatically displayed when you enter Character Edit Mode.
2. Select the **AudioMotion2** simulation character. The Motion Transfer Editor displays the settings for this character.
3. Browse to the **Strengths** tab. This page displays a list of available rig groups. You should find that the only rig group is the Standard Motion Transfer rig group.
4. Click the **Add…** button to create a new rig group. Name the new rig group **Arms Free** and click **OK**.
5. Select the **RightUpperArm**, **RightHand**, **LeftUpperArm** and **LeftHand** rig nodes. You can hold down the **Ctrl** key to select multiple rig nodes.
6. Change the rig node mode of the selected rig nodes to **NoHold**.
7. Select the **LowerSpine**, **UpperSpine**, **Head**, **RightShoulder** and **LeftShoulder** rig nodes.
8. Change the rig node mode of the selected rig nodes to **TranslationAndRotationHold**.
9. Change the rig node strength to **100.0** for all position and angular strengths.

To use the new rig group with motion transfer:

1. Select the motion transfer event on the AudioMotion2 simulation character timeline.
2. Use the Property Editor to change the **Rig Group** property of the motion transfer event to **Arms Free**.
3. Simulate the scene. You should find that the lower body of the simulation character is driven by the motion transfer event, and the upper body of

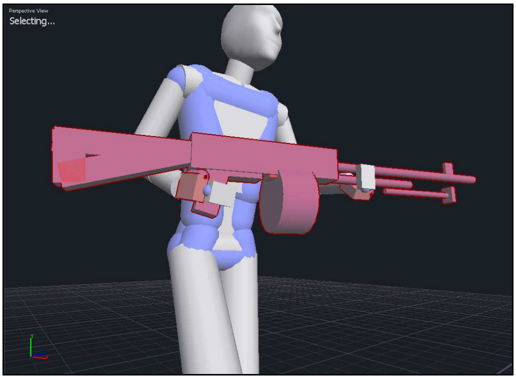
the simulation character is driven by the active pose that you applied to the upper body.

Step 8: Adding constraints between characters

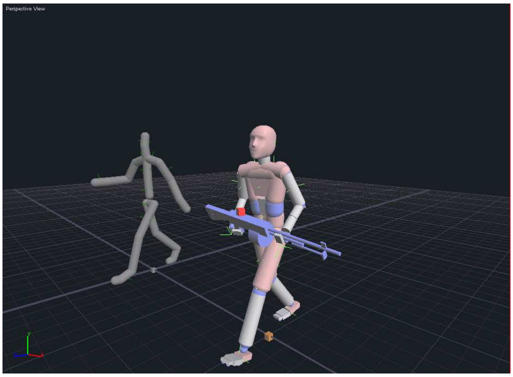
In this step we will add a **constraint** between the Gun Large prop character and the AudioMotion2 simulation character. This is required to simulate the effect of the character **holding** the rifle.

To add a constraint to the scene:

1. Right-click in any of the character timelines, and select **Create Constraint Event**. Unlike other event types, it does not matter which timeline track we use to add the constraint event.
2. Adjust the Start Frame of the new constraint event so that it starts at Frame 0. Adjust the End Frame of the new constraint event so that it matches the end frame of the animation event on the AudioMotion2_ref character timeline.
3. Select the new constraint event.
4. In the Property Editor, set its **Type** property to **JoinBodies**.
5. Locate the **[Select]** command hotlink adjacent to the **Target Objects** property, and click it. This places *endorphin* in selection mode, and allows you to specify the mass objects affected by the constraint.
6. Click on the **Gun Large** character to select its **Prop0** mass object.
7. Hold down the **Ctrl** key, and select the **LeftHandMass** and **RightHandMass** mass objects. This adds these mass objects to the list of target mass objects of the constraint event. Ensure that only these three mass objects are the *only* mass objects in the Target Objects list.



8. Simulate the scene. You should find that the simulation character now holds the rifle as it walks. You can adjust the strength of the active pose to change the degree to which its arms are driven by this active pose.



Step 9: Changing the mass of the prop character

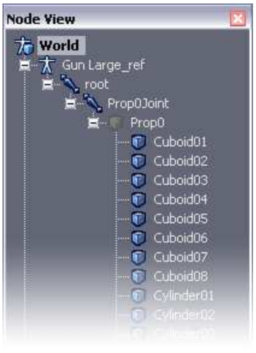
In this step we will modify the **density** of the mass object contained in the Gun Large prop character. The simulation character is holding the rifle using the strength of an active pose. We can simulate the effect of a heavier weapon by reducing the strength of the active pose, or by increasing the density or size of the mass object.

Introducing Edit Character In Scene mode

By default, you cannot edit characters directly in scenes. The usual workflow is to use Character Edit Mode to create or edit custom characters, and then add instances of these custom character into your scenes. However, it can be useful to edit characters directly within scenes themselves. Keep in mind that any changes you make to a character in a scene are **not** copied to the character definition stored in its **.nmc** file.

To modify the mass of the Gun Large character:

1. Select the **Gun Large** prop character.
2. Select **Character > Edit Character In Scene…** to edit this character directly in the scene. Alternatively, you can right-click on the character name in the Timeline Editor and select **Edit Character In Scene….** When you are editing a character directly in a scene, its timeline track is displayed in a bright red colour. In addition, the viewport label displays the text **Editing Character**.
3. Display the Node View, and expand the **Gun Large** character node hierarchy. The rifle is made up of a single mass object, **Prop0**, and a number of collision objects defining the shape of the rifle.
4. Select the **Prop0** mass object



5. In the Property Editor, set the **Density** property of the Prop0 mass object to 5.0.
6. To exit Edit Character In Scene mode, hit the **Esc** key, or right-click on the character name in the Timeline Editor and select **Exit Character Editing**.
7. Simulate the scene again. You should find that the character slumps over under the weight of the rifle.

8. Edit the Gun Large prop character again, and set its **Density** value to 10.0.
9. Simulate the scene again. You should find that the character slumps even more.
10. Experiment with different mass object densities. Also experiment with different active pose strengths. Finally, experiment with different motion transfer rig node strengths.

Conclusion

You have used motion transfer events in conjunction with active poses and constraints in order to repurpose a basic walk cycle. You have also learned about **Edit Character In Place** mode.

If you have any problems with this tutorial, open the corresponding scene **Resources\Tutorials\Tutorial 13 – Advanced Motion Transfer And Prop Interaction\Tutorial 13 – Advanced Motion Transfer And Prop Interaction - Complete.ens**. This scene is an example of how your scene should look if you have successfully followed all the steps in this tutorial.

Tutorial 14

Active animation and motion transfer

In this tutorial, we will use animation events in a special new mode—**Active Animation** mode. In this mode, we can effectively use the data in an animation event dynamically as a sequence of active poses.

Active animation events and motion transfer events are broadly similar, in the sense that a dynamically simulated character is driven (and least in part) by animation data.

However, active animation events and motion transfer events differ in a number of ways. The most important difference is that active animation events drive simulated characters **locally** via their **joints**. They cannot drive the global position or orientation of characters themselves. In contrast, motion transfer events drive simulated characters **globally** via their **mass objects**.

In addition, motion transfer events offer the ability to specify individual rig node positional and angular strengths, whereas active animation events have a single **Strength** setting that affects all its target joints equally.

In this example scene, an animated sword swipe will be used to drive both a motion transfer event, as well as an active animation event.

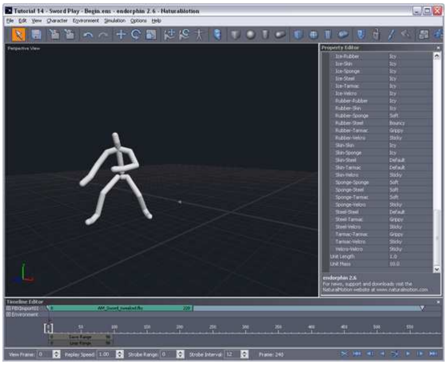
Step 1: Loading the scene

In this step, we will load a scene that contains a reference character with an animation event preloaded.

To load the scene:

1. Launch *endorphin*.
2. Select **File > Open Scene…**, and browse to select **Resources\Tutorials\Tutorial 14 - Active Animation And Motion Transfer\SwordPlay_Begin.ens**. Click **Open**.

3. **Simulate** the scene. This scene is composed of a single **AudioMotion1** reference character driven by an animation event of a sword swipe.


Tutorial 14 - Sword Play - Begin.ens - endorphin 2.6 - NaturalMotion
File Edit View Character Environment Simulation Options Help
Perspective View
Property Editor
Timeline Editor
View Frame: 0
Replay Speed: 1.00
Strobe Range: 0
Strobe Interval: 12
Frame: 240

Ice-Rubber	Icy
Ice-Skin	Icy
Ice-Sponge	Icy
Ice-Steel	Icy
Ice-Tarmac	Icy
Ice-Velcro	Icy
Rubber-Rubber	Icy
Rubber-Skin	Icy
Rubber-Sponge	Soft
Rubber-Steel	Bouncy
Rubber-Tarmac	Grippy
Rubber-Velcro	Sticky
Skin-Skin	Icy
Skin-Sponge	Icy
Skin-Steel	Default
Skin-Tarmac	Default
Skin-Velcro	Sticky
Sponge-Sponge	Soft
Sponge-Steel	Soft
Sponge-Tarmac	Soft
Sponge-Velcro	Sticky
Steel-Steel	Default
Steel-Tarmac	Grippy
Steel-Velcro	Sticky
Tarmac-Tarmac	Grippy
Tarmac-Velcro	Sticky
Velcro-Velcro	Sticky
Unit Length	1.0
Unit Mass	10.0

endorphin 2.6
For news, support and downloads visit the NaturalMotion website at www.naturalmotion.com

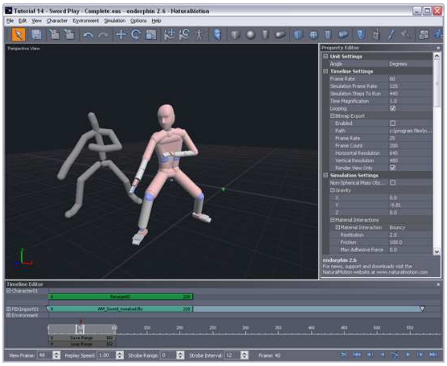
Step 2: Driving a simulation character using motion transfer

In this step, we will add a corresponding simulation character to the scene, and drive its motion using a motion transfer event.

To transfer motion to a simulation character:

1. Select **Character > Add Character…**, or click the **Add Character** button in the Main Toolbar.
2. Select the **AudioMotion1** character from the Simulation Characters list.
3. Right-click on the new **AudioMotion1** simulation character timeline in the Timeline Editor, and select **Create Motion Transfer Event**. Adjust the Start Frame so that is starts one frame after the animation event on the reference character. Adjust the End Frame to match the end frame of the animation event.
4. Select the new motion transfer event.
5. In the Property Editor, locate the **[Select]** command hotlink adjacent to the **Source Character** property. Click this hotlink to enter selection mode, and select the **Audiomotion1_ref** reference character in the viewport, Node View or Timeline Editor. **Right-click** in the viewport to leave this selection mode. The AudioMotion1_ref reference character has now been identified as the motion transfer source.

6. Simulate the scene. The **AudioMotion1_ref** reference character is driven entirely by the animation data contained in its animation event. The **AudioMotion1** simulation character is dynamically driven by the reference character via the motion transfer event. Keep in mind that the AudioMotion1 simulation character is **simulated** during the motion transfer process. This means you can modify the motion of the simulation character with other *endorphin* functionality such as behaviours and active poses.


Tutorial 14 - Sword Play - Complete.ens - endorphin 2.6 - NaturalMotion
File Edit View Character Environment Simulation Options Help
Perspective View
Property Editor
Timeline Editor
Character01
Retarget01
FBXImport01
AM_Sword_tweaked.fbx
Environment
Save Range
Loop Range

Property Editor

Unit Settings	
Angle	Degrees
Timeline Settings	
Frame Rate	60
Simulation Frame Rate	120
Simulation Steps To Run	440
Time Magnification	1.0
Looping	☑
Bitmap Export	
Enabled	☐
Path	c:\program files\n...
Frame Rate	25
Frame Count	200
Horizontal Resolution	640
Vertical Resolution	480
Render View Only	☑
Simulation Settings	
Non-Spherical Mass Obj...	☐
Gravity	
X	0.0
Y	-9.81
Z	0.0
Material Interactions	
Material Interaction	Bouncy
Restitution	2.0
Friction	100.0
Max Adhesive Force	0.0

endorphin 2.6
For news, support and downloads visit the NaturalMotion website at www.naturalmotion.com

View Frame: 40 Replay Speed: 1.00 Strobe Range: 0 Strobe Interval: 12 Frame: 40

Step 3: Preparing the initial pose

In this step, we will adjust the start pose of the simulation character. Instead of using a T-pose, it will use the pose from the first frame of the animation. This will avoid the sudden jump that you see between Frame 0 and Frame 1.

To set the initial pose of the simulation character:

1. **Simulate** for a short duration, so that at least a few frames of animation have been stored in the frame buffer, and then stop the simulation.
2. Move the Time Slider to Frame 2. The AudioMotion1 simulation character should be in the first frame of the animation.
3. Select the AudioMotion1 character, and then select **Character > Save Pose…**.
4. Save this pose as **Start.nma**. This creates an *endorphin* **.nma** pose file containing the character pose.
5. Move the Time Slider to Frame 0.
6. Ensure that the AudioMotion1 simulation character is still selected.

7. Select **Character > Load Pose…**, and browse for the **Start.nma** pose file. Do not modify any of the Load Pose settings, and click **OK**.
8. Simulate the scene. The AudioMotion1 simulation character motion should now be smooth, with no discontinuities between Frame 0 and Frame 1.

Step 4: Working with motion transfer rig groups

In this step we will create a new motion transfer **rig group**, so that the animation data only drives the lower body of the simulation character.

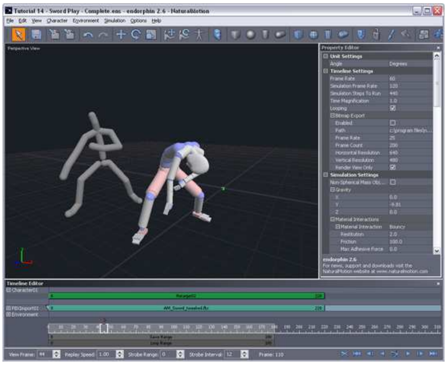
To create a new rig group:

1. Display the Motion Transfer Editor by selecting **View > Motion Transfer Editor**. The keyboard shortcut is **M**. The Motion Transfer Editor is automatically displayed when you enter Character Edit Mode.
2. Select the **AudioMotion1** simulation character. The Motion Transfer Editor displays the settings for this character.
3. Browse to the **Strengths** tab. This page displays a list of available rig groups. You should find that the only rig group is the Standard Motion Transfer rig group.
4. Click the **Add…** button to create a new rig group. Name the new rig group **Legs Only** and click **OK**.
5. Select all the rig nodes, and set the rig node mode to **NoHold**. You can select multiple rig nodes by holding down the **Ctrl** key.
6. Select the **RightUpperLeg**, **RightFoot**, **RightToes**, **LeftUpperLeg**, **LeftFoot** and **LeftToes** rig nodes, and set the rig node mode to **FullHold**.

To use the new rig group with motion transfer:

1. Select the motion transfer event on the AudioMotion1 simulation character timeline.
2. Use the Property Editor to change the **Rig Group** property of the motion transfer event to **Legs Only**.

3. Simulate the scene. You should find that the simulation character's **legs** will be driven by the motion transfer event, whereas its upper body should be loosely flopping.



Step 5: Adding an active animation event

In this step we will add an animation event to the simulation character. This event will contain the same sword swipe animation data used in the reference character's animation event. However, the new animation event will be used as an **active** animation event applied directly to the simulation character.

To add an active animation event:

1. Select the **Audiomotion1** simulation character.
2. Select **File > Import**, or click the **Import** button in the Main Toolbar. The keyboard shortcut is **I**.
3. Browse to select the **SwordSwipe.fbx** file, and click **OK**.
4. Select the **Hips** node as the import root node in the node tree.
5. In the Import Options dialog, turn off **Create Simulation Events**. When this setting is turned off, the character simulation mode is unchanged. In our case, this means the character will remain in Full Simulation mode, and so the animation will drive this fully-simulated character **actively**. In contrast, when simulation events are added to set the simulation mode to No Simulation, Collision Only or Collision With Momentum, the animation drives the character **passively**.
6. Turn on the **Transfer From Reference Character** setting, and browse to select the AudioMotion1 reference character. Keep the other settings at their default values, and click **OK**.

7. Animation will be imported from the **SwordSwipe.fbx** file using Auto Motion Transfer. A new animation event will be created.

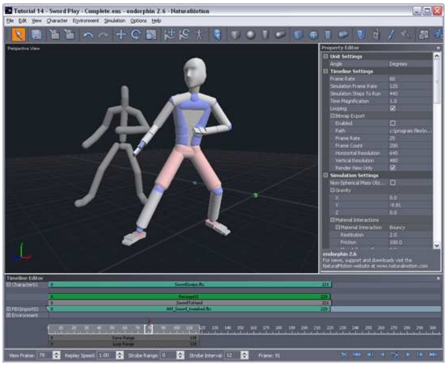
   Note We now have **two** animation events in the scene, **both** using the **SwordSwipe.fbx** animation data. One of these animation events is applied to the **reference** character, and was created by direct animation import. The other animation event is applied to the **simulation** character, and was created using Auto Motion Transfer from the reference character.

8. Ensure that the new animation event has its Start Frame set to Frame 0.

9. Select the animation event and drag it upwards until it is placed on the **topmost** timeline track. This ensures that this new animation event has the highest priority in the simulation character timeline.

10. Use the Property Editor to set the **Strength** property of the new animation event to 1.0.

11. Use the Property Editor to turn on the **Active** property of the new animation event. This is the critical step that converts the event from a standard, or passive, animation event, into an active animation event.



Unit Settings	
Angle	Degrees
Timeline Settings	
Frame Rate	60
Simulation Frame Rate	120
Simulation Steps To Run	440
Time Magnification	1.0
Looping	✓
Bitmap Export	
Enabled	☐
Path	c:\program files\n...
Frame Rate	25
Frame Count	200
Horizontal Resolution	640
Vertical Resolution	480
Render View Only	✓
Simulation Settings	
Non-Spherical Mass Obj...	☐
Gravity	
X	0.0
Y	-9.81
Z	0.0
Material Interactions	
Material Interaction	Bouncy
Restitution	2.0
Friction	100.0

endorphin 2.6
For news, support and downloads visit the NaturalMotion website at www.naturalmotion.com

View Frame: 78 Replay Speed: 1.00 Strobe Range: 0 Strobe Interval: 12 Frame: 91

12. Simulate the scene. The same animation source file is now driving two different sections of the simulation character using two different routes. Each of these routes has a different effect:

    - The **motion transfer event** drives the lower body of the simulation character using the animation event on the reference character timeline. This event drives the mass objects of the legs in global coordinates.

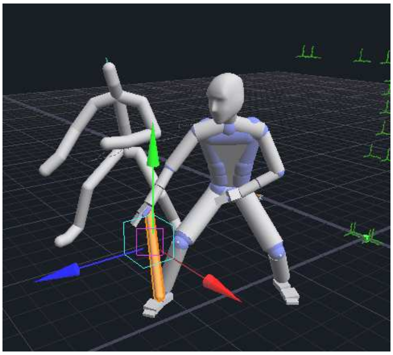
    - The **active animation event** drives the upper body of the simulation character using the data in the event. This event drives the joints of the torso and arms in local coordinates.

Step 6: Connecting props to the simulation character

In this step we will add some mass objects to the simulation character. These objects will model the effect of weapons held by the simulation character.

To add a prop mass object to the Environment character:

1. Select **Environment > Create Sphyl Mass Object**. Alternatively, click the **Create Sphyl Mass Object** button in the Main Toolbar. A new sphyl mass object is added to the Environment character.
2. Select the new mass object. Using the Property Editor, set the mass object **Length** property to 0.8. Also, set the **Radius** property to 0.02.
3. Use the Move and Rotate tools to position and rotate the mass object so that one end of the object is close to the right hand of the simulation character. Alternatively, enter the following values directly as the Position And Orientation properties of the mass object:
    - **Position: X:** 0.22; **Y:** 0.70; **Z:** 2.43.
    - **Orientation: X:** -44.0; **Y:** +46.0; **Z:** -64.0.



To connect the mass object to the simulation character:

1. Right-click on the AudioMotion1 simulation character timeline in the Timeline Editor and select **Create Constraint Event**.
2. Adjust the Start Frame and End Frame properties of the new constrain event to match the range of the animation events.
3. Use the Property Editor to set the constraint event **Type** to JoinBodies.

4. Locate the **[Select]** command hotlink adjacent to the **Target Objects** property, and click it. This places *endorphin* in selection mode, and allows you to specify the mass objects affected by the constraint.
5. Select the new mass object.
6. Hold down the **Ctrl** key and select the **RightHandMass** mass object.
7. Simulate the scene. You should find that the character now holds the mass object during the sword swing.

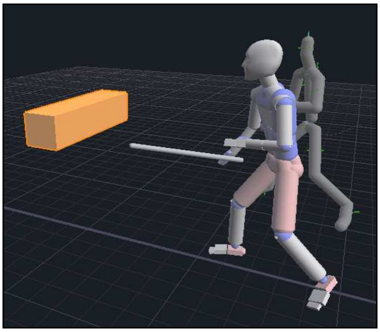
Step 7: Adding collision objects to the environment

In this step, we will add a collision object to the Environment character. This object represents an object that the simulation character may be striking.

To add a collision object to the Environment character:

1. Ensure no objects are selected. The Property Editor should be displaying the Simulation Settings and Timeline Settings.
2. Create a new box collision object in the Environment character by selecting **Environment > Create Box Collision Object**, or by clicking the **Create Box Collision Object** button on the Main Toolbar.
3. Select the new collision object.
4. Use the Property Editor to set its **Width** value to 1.0. Also, set its **Height** and **Length** values to 0.25.

4. Use the Move and Rotate tools to position the new collision object in front of the simulation character. Alternatively, enter the following values directly as the Position And Orientation properties of the collision object:
    - **Position: X:** 0.0; **Y:** 1.08; **Z:** 3.10.
    - **Orientation: X:** 0.0; **Y:** 0.0; **Z:** 0.0.



Step 8: Modifying the mass object density

In this step, we will change the density of the mass object representing the weapon held by the simulation character.

To modify the density of the mass object:

1. Select the **Prop01** mass object.
2. Use the Property Editor to increase its **Density** value to 10.0.
3. Simulate the scene. You should see the right arm of the simulation character swing more **slowly** than its corresponding source animation data. This is due to the inertia of the heavier mass object.
4. Experiment with different density values.

Step 9: Modifying the active animation event strength

In this step, we will modify the strength of the active animation event. The strength of an active animation event is very similar to the strength of an active pose event. This is because active animation events are effectively just a **sequence of active poses**.

To modify the strength of the active animation event:

1. Select the active animation event on **AudioMotion1** simulation character timeline.
2. Use the Property Editor to set its **Strength** value to 0.3.
3. Simulate the scene. You should find the simulation character slumping and unable to raise the mass object to its previous height.
4. Experiment with different active animation strength values.

Step 10: Modifying the collision object material

In this step, we will modify the **material** of the collision object. Materials specify the surface friction properties of an object. For example, you can specify a collision object as being smooth or rough. Materials also specify the volumetric properties of an object. For example, you can specify whether an object is rigid, soft or rubbery.

To change the material of the collision object:

1. Select the active animation event on the AudioMotion1 simulation character timeline, and set its **Strength** value back to 1.0.
2. Select the box collision object created in Step 7.
3. Use the Property Editor to set its **Material** property to **Rubber**.
4. Simulate the scene. You should find that the mass object now **bounces** off the collision object after colliding with it. Also note that this bounce affects the simulation character itself, in addition to the mass object.
5. Experiment with different animation strength values and materials to modify the collision response.
6. Use the Property Editor to set the **Material** property of the collision object to **Sponge**.
7. Simulate the scene. You should now find that the mass object penetrates the collision object after colliding with it.

Step 11: Adding a helper constraint

In this step, we will add a short-duration constraint event. This constraint will lock the mass object for a few frames, in order to mimic the effect of the mass object becoming temporarily lodged inside the collision object.

To add a help constraint event:

1. Right-click the AudioMotion1 simulation character name in the Timeline Editor, and select **Add Event Track**. This adds a new event track to the character's timeline.
2. Simulate the scene, in order to fill up the frame buffer.
3. Move the Time Slider to the frame at which the mass object has its deepest penetration into the collision object.
4. Right-click on the simulation character timeline at this frame, and select **Create Constraint Event**. You will automatically be placed into selection mode. This mode allows you to specify the mass objects that are to be constrained by the constraint event. When you are in the selection mode, *endorphin* displays the text **Selecting…** in the viewport label.
5. Select the sword mass object.
6. Modify the constraint event End Frame so that the event has a duration of around 45 frames. For example, it might have a Start Frame of Frame 50, and an End Frame of Frame 95.
7. Simulate the scene. You should find that the sword mass object now penetrates into the collision object, and is held in place for the duration of the constraint event. When the constraint event ends, the mass object springs back and the character quickly resumes its original animation.

Conclusion

You have learned to use **active animation events poses**, which effectively influence the motion of simulated characters via a sequence of single-frame active pose events. You have also used helper constraints to increase the realism of simulated scenes.

If you have any problems with this tutorial, open the corresponding scene Resources\Tutorials\Tutorial 14 – Active Animation And Motion Transfer\Tutorial 14 – Active Animation And Motion Transfer - Complete.ens. This scene is an example of how your scene should look if you have successfully followed all the steps in this tutorial.

Tutorial 15

Working with fluids

In this tutorial we will concentrate on learning how to use a new *endorphin* 2.7 feature: **fluids**.

To model the effect of fluids, **fluid objects** are added to a character, and a **Fluid** behaviour is applied to the character. The Fluid behaviour generates drag and buoyancy forces on the fluid objects. Drag forces have the effect of slowing down fluid objects as they move through the fluid. Buoyancy forces have the effect of causing fluid objects to float in the fluid.

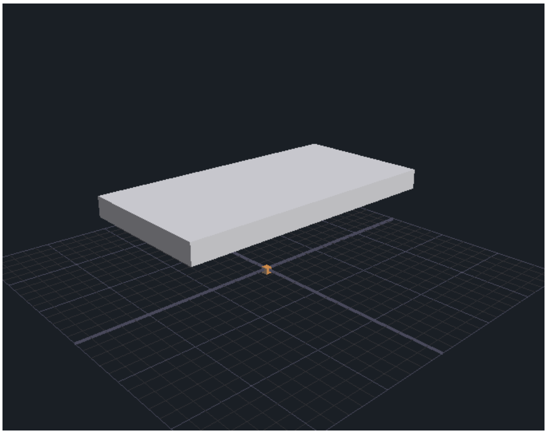
In this example scene, you will create a small boat and a character that will dive from the boat into water.

Step 1: Building a simple boat using fluid objects

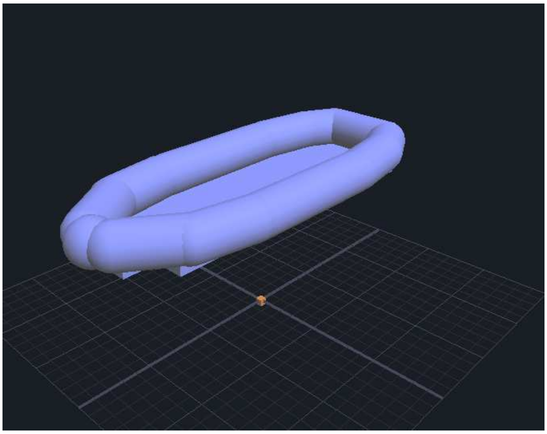
In this step we will create a simple boat that can interact with *endorphin* fluids.

To create a new boat character:

1. Launch *endorphin*, and select **File > New Character…**.
2. Select the **Standard Prop Character** from the standard character list, and click **OK**.
3. Reshape the mass object to roughly represent the size and weigh of a small boat.



4. Build the collision surface of the boat using sphyl collision objects for the sides and cube collision objects for the bottom.



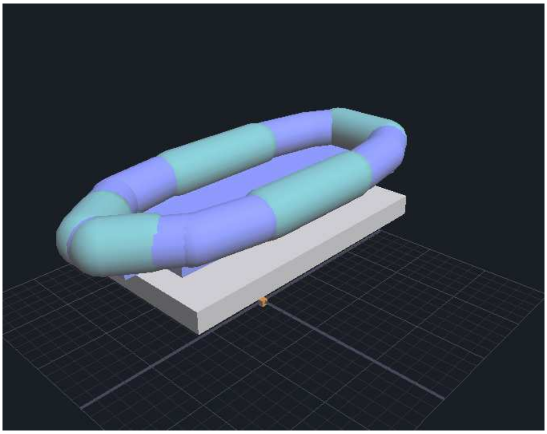
To model the boat buoyancy using fluid objects:

1. Select the boat’s mass object.
2. Select **Character > Create Sphyl Fluid Object**. This will create a new fluid object that represents a floating body. The default fluid object shape is a sphyl but this can be changed using the Property Editor.
3. Use the **Move**, **Rotate** and **Scale** tools to position and resize the fluid object to match one of the side collision objects.
4. Keep creating as many fluid objects as you need to define the parts of the boat that will interact with the fluid. Fluid objects do not need to cover every part of your object—only those that are going to be affected by the fluid.

As a general rule, try to keep the number of fluid objects to a minimum. Also, try not to overlap fluid objects where possible. Buoyancy is calculated based upon the total volume of the fluid objects, and does not correct for intersecting fluid objects. This means that the volume of intersection is effectively counted twice when calculating the total buoyancy force.

5. Once you are happy with the boat character, select **File > Save Character**, and save the prop character as **MyBoat**. Click **OK**.

   Hint To increase the stability of the boat, try moving the mass object downwards with respect to the centre of mass of the boat.



Step 2: Adding characters to the scene

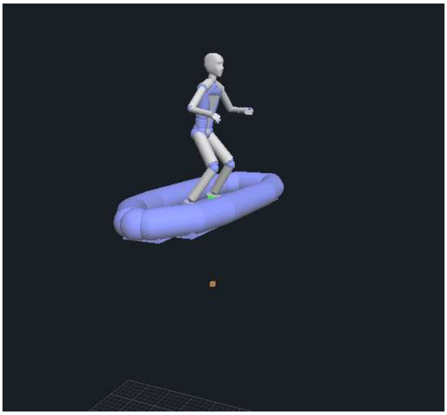
In this step we will create a new scene, and add the boat character to it. We will add an instance of the new standard simulation character that ships with *endorphin* 2.7. This character has already been equipped with fluid objects.

To load the boat prop character in the scene:

1. Select **File > New Scene**.
2. Load your small boat character. To do this select **Character > Add Character**. Find the prop character named **MyBoat** and click **OK**.

To load and position the new standard simulation character in the scene:

1. Select the **Character01** standard simulation character using the Timeline Editor or Node View.
2. Use the **Move** tool to move the character so that it is placed roughly above the small boat.
3. We now apply a pose to the character. With the character still selected, select **Character > Load Pose**. Browse to the resource folder for this tutorial and select the pose file named **on_the_dingy.nma**. (A dingy is type of small boat.)
4. Click **OK** in the Load Pose window. The character will be placed quite high in the scene—you may need to adjust the viewport zoom to see it.
5. Select the boat and use the **Move** tool to move it into position underneath the character.



Step 3: Adding a fluid cube to the environment

In this step we will add a fluid cube to the scene that will constitute the volume of the fluid. We will then set some of the cube’s properties.

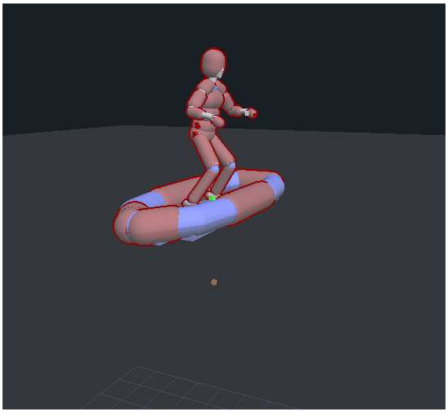
To add a fluid cube:

1. Right-click on the simulation character timeline, and select **Create Behaviour Event** to create a new behaviour event for that character.

2. Set the behaviour Name property to **Physical Effect: Fluid 1**. Each fluid behaviour has a corresponding **fluid cube**. Most often, you will use fluid cubes to represent water, but you can represent thicker fluids like quicksand and mud, as well as thinner fluids like air. Note that the fluid cube is not displayed in the viewport until you have simulated at least once.
3. Edit the Start Frame and End Frame properties of the Fluid behaviour, so that it begins at Frame 0 and ends at Frame 600.

To edit the fluid cube properties:

1. Right click on the viewport and select **Fluid Objects**. Fluid objects should now be displayed.
2. Fluid behaviours include a number of parameters that define the properties of the fluid contained by the fluid cube, as well as a **Floating Bodies** property that defines the set of fluid objects that interact with the fluid cube. Click the **[Select]** hotlink in the Property Editor to edit the Floating Bodies selection set.
3. Use box-selection to select all objects in the viewport. All selected fluid objects are added to the Floating Bodies set of the Fluid behaviour.
4. Locate the **Cube Size** parameter, and change value of the cube size to 20. This will increase the size of the fluid cube.
5. Keep all other parameters at their default settings. The default fluid density matches that of water.
6. **Simulate** the scene to see how the fluid behaviour behaves. The fluid cube should be displayed as a semi-transparent cube, and the simulation character and the boat prop character should float on the surface of the water.



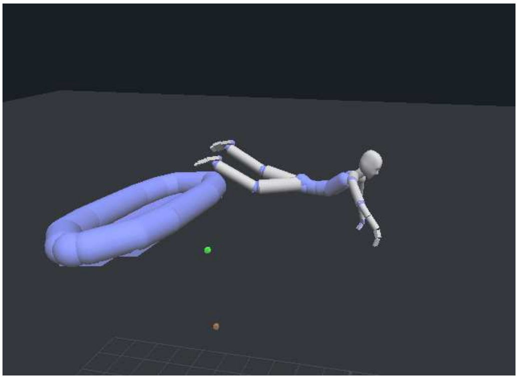
Step 4: Completing the scene

In this step we will place the boat prop character and the simulation character above the surface of the fluid, and add a Jump And Dive behaviour to the character. This will make the character dive into the fluid cube.

You will also experiment with other parameters of the fluid and refine your scene so that the final animation is richer and more compelling.

To add a dive behaviour to the character:

1. Right-click on the standard simulation character in the Timeline Editor and select **Create Behaviour Event**.
2. Set the behaviour Name property to be the **Jump And Dive 1**.
3. Adjust the timing of this behaviour so that it begins at Frame 0 and ends at Frame 150.
4. **Simulate** the scene. The first jump might be unsuccessful. Try experimenting with the jump timing and strength parameters to obtain a clean and successful jump.



To refine the diving motion:

1. Try to add behaviours to the character so that it reacts while underneath the water. For example, try adding **Arms Windmill 2** and **Legs Kick 2** behaviours to the character—these behaviour both generate the kinds of movement that one might expect from the submerged character. Experiment with the behaviour parameters to obtain a better motion.
2. To obtain a smoother and more plausible trajectory of the diving character within the fluid, try changing the **drag** parameters of the fluid behaviour. The drag parameters define the degree to which the motion of fluid objects is retarded by the fluid. For example, try decreasing the value of the **Drag** parameter and increasing the value of the **Angular Drag** parameter. Also, try setting the value of **Turbulent Drag** to 0. Keep simulating the scene and changing these parameters until you are satisfied with the resulting motion.

   Hint For better scene visualisation, right-click on each character in the **Node View** and select **Hide Type > Fluid Objects**. You can also hide the mass object for the small boat prop character by selecting this mass object in the viewport and clicking on the correspondent icon in the Node View to hide it.

Further experiments

- Try posing the **head** of the character so that it remains aligned with its body. Introduce some soft constraints to the chest of the character to help increase its stability while its head remains above the level of the fluid cube.
- Try changing the **size**, **shape** and **position** of the fluid objects for both the small boat and the simulation character to see how these changes affect the buoyancy of the characters. Experiment with the various fluid parameters such its **density** and **stream vector**. You can visualise the stream vector as a current or wind direction.
- Try turning on the fluid **waving system**. Increase or decrease the **amplitude** and **frequency** of the waves.
- Try creating your own prop characters and attach fluid objects to them. See if you can correctly model their expected buoyancy properties.
- Try simulating **multiple** fluid cubes in the same scene to model different fluid effects.

Conclusion

In this tutorial we have looked at how to work with the **fluid** physical effect behaviour.

We have learned how to model characters using fluid objects, and how to add a fluid cube in a scene. We have also modified the properties of the cube to obtain different fluid effects and how to control the way objects are affected by the fluid.

If you have any problems with this tutorial, open the corresponding scene **Resources\Tutorials\Tutorial 15 – Working With Fluids\Tutorial 15 – Working With Fluids - Complete.ens**. This scene is an example of how your scene should look if you have successfully followed all the steps in this tutorial.

Tutorial 16

Using physical effects

In this tutorial we will introduce a range of **physical effect** behaviours. Unlike most behaviours—which act on characters internally by driving their joints—physical effect behaviours model environmental conditions that influence characters externally.

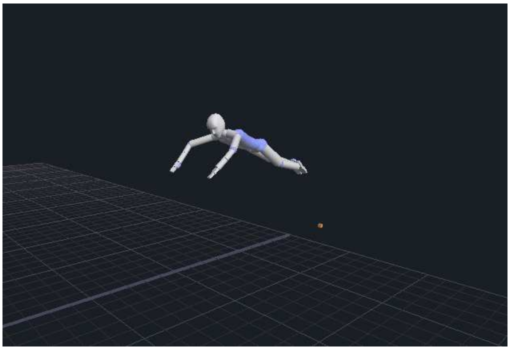
You will create a scene in which two simulation characters engage in aerial combat. Physical effects—such as pushing, twisting and damping behaviours—will be used to generate various types of motion.

Step 1: Creating a stylised jumping character

In this step we will add a character to a scene, and use physical effect behaviours to create a highly-stylised jump that will also involve a twist.

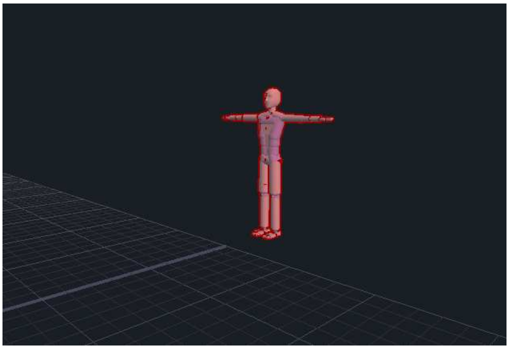
To position the character and start the jump:

1. Select **File > New Scene**, and select the **Character01** standard simulation character using the Timeline Editor.
2. Use the **Move** tool to move the character back to the edge of the grid. This is a Z-position of approximately -10.5.
3. **Right-click** on the Character01 timeline and select **Create Behaviour Event**.
4. Set the new behaviour name to **Jump And Dive 1**.
5. Move the behaviour marker so it lies on the **topmost** timeline track.
6. Adjust the behaviour timing so that it begins at Frame 0 and ends at Frame 80.
7. Simulate the scene to obtain some visual feedback.



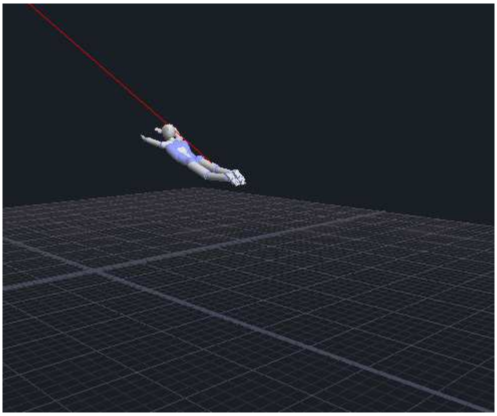
To create a long jump using the Push physical effect:

1. Right-click on the Character01 timeline and select **Create Behaviour Event**.
2. Set the new behaviour name to **Physical Effect: Push 1**.
3. Move the behaviour marker so that is on the **middle** timeline track. This ensures that this behaviour has lower **priority** than the Jump And Dive 1 behaviour.
4. Adjust the behaviour timing so that it begins at Frame 67, and ends at Frame 117.
5. The Force X, Force Y and Force Z properties define the direction and strength of the force applied by this behaviour. Set the **Force Y** property to 20. Set the **Force Z** property to 40.
6. The **TargetMassObjects** property defined the mass objects that have the force applied to them. Note that the usual Target Objects property is **not** used by this behaviour. Click the **[Select]** hotlink to edit the target mass objects.
7. Use box-selection to select the entire character in the viewport. Make sure that all its mass objects have been selected.
8. Right-click in the viewport to exit **Selection** mode.
9. Simulate the scene. The character should now be pushed forward rapidly by the Push behaviour. You should also notice a **red arrow** appearing in the viewport while the physical effect is simulated. This arrow represents the magnitude and direction of the push force.



To stabilise the jump with an active pose:

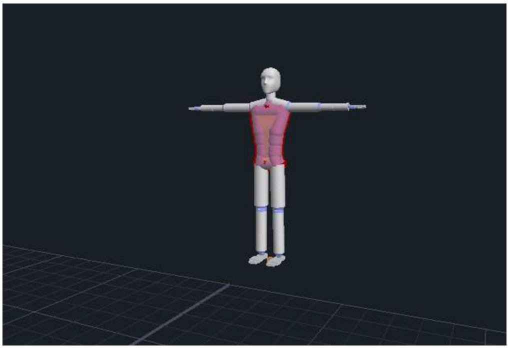
1. To make the jump more stable we will apply an **active pose** to the character in order to increase its stiffness. Move the Time Slider to Frame 83.
2. Right-click on the Character01 timeline, and select **Create Active Pose Event**.
3. Move the active pose marker to the **bottom** timeline track.
4. Adjust the active pose timing so that it begins at Frame 83 and ends at Frame 117.
5. In the Property Editor, set the active pose **Strength** property to 15.
6. Simulate the scene. You should now see a more stable jump.



To add a twist during the jump:

1. **Right-click** on the character name section of the **Character01** timeline and select **Add Event Track**.
2. **Right-click** on the Character01 timeline and select **Create Behaviour Event**.
3. Set the new behaviour name to **Physical Effect: Twist 1**.

4. Move the behaviour marker so that is on the **bottom** timeline track. This ensures that this behaviour has lower **priority** than the active pose.
5. Adjust the behaviour timing so that it begins at Frame 70, and ends at Frame 117.
6. The Torque X, Torque Y and Torque Z properties define the axis about which the torque force is applied, and its strength. Set the **Torque Z** property to 15. This will ensure that objects are rotated about the Z-axis by the Twist force.
7. The **TargetMetaBodies** property defined the mass objects that have the torque applied to them. Note that the usual Target Objects property is **not** used by this behaviour. Click the **[Select]** hotlink to edit the target mass objects.
8. Zoom into the character torso, and select the four body parts formed by the pelvis and the three spine objects.
9. Right-click in the viewport to exit **Selection** mode.



10. Simulate the scene. You should now have obtained a Twist jump. Next we will add another character to the scene, and have this character jump towards our existing character and collide with it in midair.

    Note You can turn off the display of behaviour helper graphics by right-clicking on the viewport, and turning off the **Display > Events** setting. Also, if the behaviour helper graphics are too large or small you can change their size by adjusting the **Arrow Scale** property of the Push and Twist behaviours.

Step 2: Creating an additional jumping character

In this step we will introduce a new character available from our custom character folder. We will try to make this character jump towards our first character from the opposite side while assuming a fighting pose.

To position a new character:

1. Add a new character by selecting **Character > Add Character**.
2. Browse to the **Custom** character folder (**[install location]\Resources\Characters\Custom**) and select the **Muscular** simulation character. Click **OK**.
3. Use the **Move** tool to position this new character at the opposite side of the grid to the previously created character.
4. While the new character is still selected, rotate it around the Y-axis by setting its **Y Orientation** property to 180 in the Property Edutor. The two characters should now be facing each other.

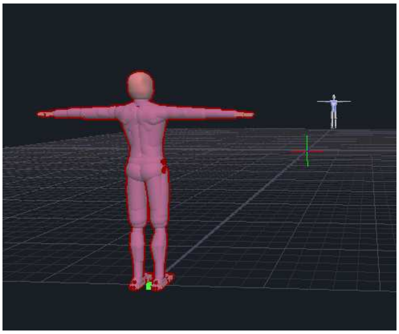
To start a new jump:

1. Right-click on the **Muscular** character timeline, and select **Create Behaviour Event**. Set the behaviour name to **Jump And Dive 1**.
2. Adjust the new behaviour timing so that it begins at Frame 0 and ends at Frame 60.
3. Move the behaviour marker to the **topmost** timeline track.

To create a new long jump:

1. Right-click on the **Muscular** character timeline on the middle track and select **Create Behaviour Event**. Set the behaviour name to **Physical Effect: Push 1**.
2. Adjust the new behaviour timing so that it begins at Frame 54 and ends at Frame 117.
3. Move the behaviour marker to the **middle** timeline track.
4. Set the behaviour **Force Y** property to 10 and the **Force Z** property to -30.
5. Click the **[Select]** hotlink of the **TargetMetaBodies** property, and select the mass objects that which will be affected by the Push behaviour. Use box-selection to select the entire character in the viewport. Make sure that all objects have been selected.

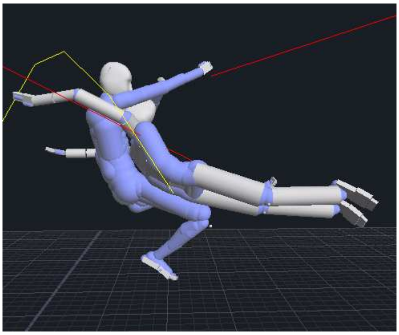
6. Right-click in the viewport to exit Selection mode.
7. Simulate the scene for visual feedback.



To specify a combat pose for the new character:

For this tutorial we have provided you with a **pose** file for you to use with the new character. This pose will be assigned to the character as an active pose.

1. Right-click on the **Muscular** character timeline and select **Create Active Pose Event**.
2. Move the active pose marker to the **bottom** timeline track.
3. Right-click on the active pose marker and select **Load Pose**.
4. Browse to the **fighting_pose.nma** file located in the resource folder for this tutorial. Double-click on the pose file to load it.
5. Adjust the active pose timing so that it begins at Frame 75 and ends at Frame 135.
6. Set the active pose **Strength** parameter to 5.
7. Set the active pose **Blend Period** to 0.7.
8. Simulate the scene for visual feedback. The characters should now collide in flight. In the next step we will concentrate on the collision between the characters.



Note At this stage, the characters are moving extremely fast. You may find that some intersections between collision objects are not detected at the standard simulation frame rate, resulting in undesirable penetration of collision objects. In later steps we will reduce the velocities of the characters to avoid this problem.

Step 3: Creating a dynamic collision between the characters

In this step we will try to obtain a spectacular impact between the two characters so that one of the characters the standard simulation character will try and grab the other character while in mid-air.

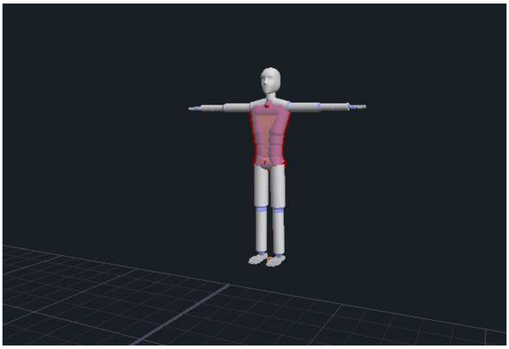
For this purpose we will modify some of the parameters for our jump events. We will also introduce the **damping** physical effect. The damping effect lets you slow down the motion of a character.

To make the characters collide centrally:

1. Select the **Push** event on the **Muscular** character timeline and adjust its **Force Y** property using the Property Editor. Experiment with slightly higher and lower values of Force Y. The goal is to have the simulation character narrowly miss the kick of the muscular character. Simulate the scene each time until you are satisfied with how the two characters intersect during the collision.
2. Use the **Move** and **Rotate** tools to slightly adjust the position and orientation of the characters in order to get the arms of the twisting character around the body of the other character. Experiment with increasing the distance between the two characters. Again, continue making small changes iteratively until you are satisfied with the collision of the two characters.

To slow down the standard character:

1. Right-click on the Character01 standard character timeline, and select **Create Behaviour Event**.
2. Set the behaviour name to **Physical Effect: Damping 2**.
3. Move the behaviour marker to the **bottom** timeline track. If necessary, create an additional timeline marker track.
4. Adjust the behaviour timing so that it begins two frames before the collision between the two characters. Resize the event so that its duration is 10 frames.
5. With the event still selected, use the Property Editor to set the **LinearVelocityDamping** property to 0, and set the **AngularVelocityDamping** property to 30. These parameters define the magnitudes at which the linear and angular velocities should be reduced.
6. Click the **[Select]** hotlink in the **MassObjects** section.
7. Adjust the viewport camera so that it focuses on the standard character.
8. Select the pelvis mass object together with the three mass objects that form the spine of the character.
9. Right-click in the viewport to exit Selection mode.

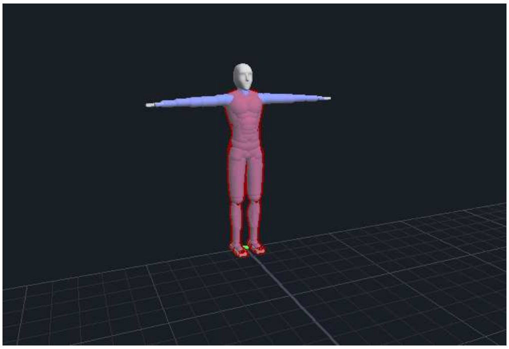


10. Adjust the **End Frame** properties for both the Twist and Push physical effects and the active pose present on the standard character timeline so that they match the **Start Frame** property of the damping physical effect.

To slow down the muscular character:

1. Right-click on the muscular character timeline, and select **Create Behaviour Event**.

2. Set the behaviour name to **Physical Effect: Damping 2**, and move the behaviour marker to the **bottom** timeline track. If necessary, create an additional timeline marker track.
3. Adjust the behaviour timing so that it begins two frames before the collision between the two characters. Resize the event so that its duration is 6 frames.
4. With the event still selected, use the Property Editor to set the **LinearVelocityDamping** property to 40. Leave the **AngularVelocityDamping** property at its default value.
5. Click the **[Select]** hotlink in the **MassObjects** section.
6. Adjust the viewport camera so that it focuses on the muscular character.
7. Select the entire character, excluding its shoulders, arms and head.
8. Right-click in the viewport to exit Selection mode.

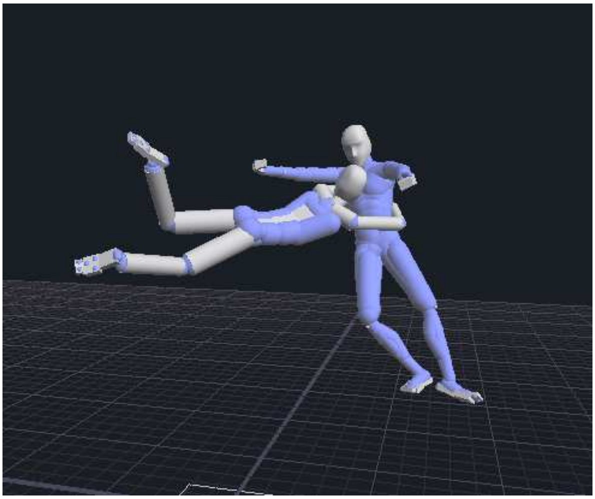


9. Simulate the scene. Note that the two characters suddenly decrease their speed when the damping physical effects are triggered.

To make the standard character grab the muscular character:

1. Right-click on the standard character timeline and select **Create Behaviour Event**.
2. Set the behaviour name to **Tackle 5**.
3. Adjust the timing of the behaviour so that it begins the same as the damping physical effect.
4. With the event selected, set its **Arms Strength** and **Arms Speed** properties both to 1.
5. Click the **[Select]** hotlink in the **TargetUpSpine** section. While in **Selection** mode, select the **UpperSpineMass** object of the muscle character. Right-click in the viewport to exit Selection mode.

6. Click the **[Select]** hotlink in the **TargetPelvis** section. While in **Selection** mode, select the **PelvisMass** object of the muscle character. Right-click in the viewport to exit Selection mode.
7. Simulate the scene. The standard character should now be able to grab the muscular character while it is in mid-air. If this is not the case, continue to adjust the parameters of the **damping** physical effect for both characters until their velocities are slow enough to allow the activation of the tackle behaviour.



Hint Check that the standard character active pose event does not extend into the timeframe of the tackle behaviour, as the active pose stiffness can override the behaviour and prevent it from simulating correctly.

Further experiments

- Try changing the **poses** used by the active pose events, as well as the attack angles, to obtain different attack results.
- Try experimenting with the various physical effects by applying them to different body parts and modifying their related parameters.

Conclusion

In this tutorial we have looked at how to work with the **Push** and **Twist** physical effect behaviours to create linear and rotational forces that apply to characters over multiple frames. We have also used the **damping** behaviour to slow down to motion of various body parts.

We have also looked at how behaviours and active poses can be used in conjunction with the physical effect behaviours to create more compelling animation.

If you have any problems with this tutorial, open the corresponding scene **Resources\Tutorials\Tutorial 16 - Using Physical Effects\Tutorial 16 – Using Physical Effects - Complete.ens**. This scene is an example of how your scene should look if you have successfully followed all the steps in this tutorial.

Tutorial 17

Working with Maya

You can use *endorphin* with **Maya** in an animation pipeline.

To do so, you need to configure the pipeline. You can then import animation data from Maya into *endorphin*, and export animation data back from *endorphin* into Maya. This pipeline requires the use **FBX** for transferring data in both directions.

The following six steps outline the process of transferring data from Maya to *endorphin* and back, using specific example files.

Step 1: Preparing your Maya character

In this step you will prepare your Maya character so that it can be used in *endorphin*.

You need to ensure the following:

- Ensure that your Maya character is posed in a similar T-pose to the standard *endorphin* character. Your character's legs should point straight down, and your character's arms should point straight out in a **sideways** direction.
- Ideally, you should set the Maya Linear Working Units to **Centimeters**. However, if this is not possible, you can still make scale adjustments during import and export.

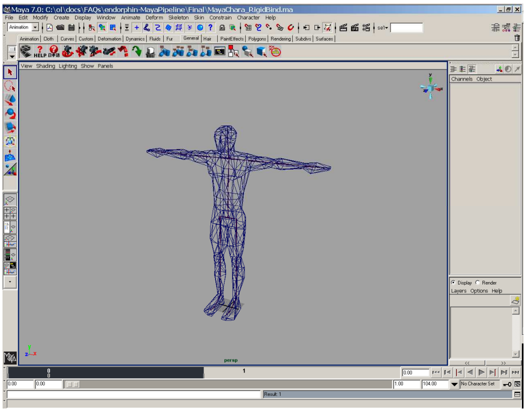
Step 2: Exporting your character from Maya

In this step you will export your character as an **FBX** file. You will also save the character skin mesh as an **OBJ** mesh file.

To save your Maya to FBX and OBJ files:

1. Launch Maya.

2. Open the Maya scene **Resources\Tutorials\Tutorial 17 – Working with Maya\MayaChara_RigidBind.ma**.
3. Select the character in this scene.
4. Select **File** > **Export All**. Choose **FBX** from the File Type list, and specify **MayaChara** as the export filename.
5. Click **Export**. The skeletal structure of your Maya character will be saved into an FBX file.
6. Select **File** > **Export All**. Choose **OBJ** from the File Type list, and specify **MayaChara_Mesh** as the export filename. It is generally good working practice to name the FBX and OBJ files using the same name.
7. Click **Export**. The polygonal skin mesh of the Maya character will be saved into an OBJ file.


Maya 7.0: C:\ol\docs\FAQs\endorphin-MayaPipeline\Final\MayaChara_RigidBind.ma
File Edit Modify Create Display Window Animate Deform Skeleton Skin Constrain Character Help
View Shading Lighting Show Panels
Channels Object
Display Render
Layers Options Help
persp
No Character Set
Result: 1

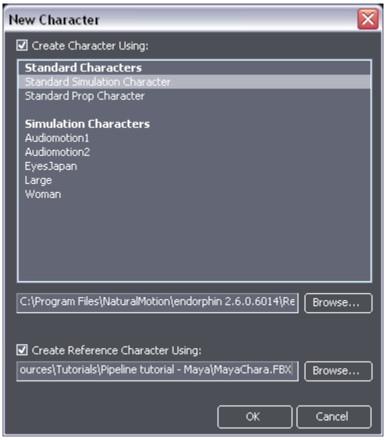
Step 3: Creating corresponding *endorphin* characters

In this step, you will create a **pair** of characters in *endorphin* that represent your Maya character. Once you have created this simulation-reference character pair, you will be able to use them to import animation from Maya into *endorphin* using dynamic motion transfer.

One of the characters will be an *endorphin* **simulation** character that has been reshaped to have the same dimensions as your Maya character. The other character will be an *endorphin* **reference** character created from the FBX file you saved in Step 2.

To create a simulation-reference character pair:

1. Select **File > New Character…**.
2. Choose the **Standard Simulation Character** as the simulation character.
3. Turn on the **Reference Character** setting, and import the **MayaChara.FBX** file that you saved in Step 2. In the **Import Options** dialog, select the root of your FBX character from the joint hierarchy tree, and then click **OK**.


New Character

☑ Create Character Using:

Standard Characters
Standard Simulation Character
Standard Prop Character

Simulation Characters
Audiomotion1
Audiomotion2
EyesJapan
Large
Woman

C:\Program Files\NaturalMotion\endorphin 2.6.0.6014\Re

☑ Create Reference Character Using:

ources\Tutorials\Pipeline tutorial - Maya\MayaChara.FBX

4. You will automatically enter **Character Edit Mode**. In this mode you will be able to edit both the simulation and the reference characters. Turn on the **Skeletal View** viewport display setting, and then **snap align** the simulation character joints to match the reference character joints. Also, adjust the sizes, shapes and positions of the mass objects and collision objects of the simulation character, if needed. See the User Guide and tutorials for more detailed information on creating simulation-reference character pairs.
5. Display the **Motion Transfer Editor**, and use the following connection settings to connect the mass objects of your reference character to the **Standard Character Rig**. See the User Guide and tutorials for more detailed information on rigging reference characters.
6. Once the simulation and reference character skeletons are aligned, select **File > Save Character…**. You can browse for a location to save the character, and specify **MayaChara** as the character name. *endorphin* will save both the simulation and reference characters as two separate **.nmc** character files.

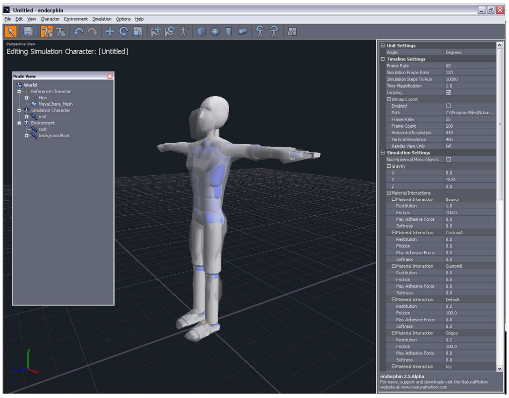
Keep in mind that the reference character will be used to **import** and **export** animation, whereas the simulation character will be used to actually generate new animation data in *endorphin*.

Step 4: Filling out the simulation character

In this step, you will modify the collision objects of the simulation character to fill the volume of the skin mesh saved in the OBJ file in Step 2. You may also add new collision objects as well as modifying existing collision objects. This is often called **filling out** a character.

To fill out the simulation character:

1. While you are still in Character Edit Mode, right-click and select **Shaded View**.
2. Click **Activate Reference Character** button on the Main Toolbar. This activates the reference character.
3. Select **File > Import** and select the **MayaChara_Mesh.OBJ** file that you saved from Maya in Step 2. Click **Open**, and then click the **OK** button on the Import Options dialog. If necessary, you may need to adjust the orientation of the OBJ file in the Import Options dialog.
4. The OBJ file is imported as a new graphical object, and associated with the reference character. The graphical object is displayed partially transparently.
5. Select **View > Node View** to display the Node View. The keyboard shortcut is **N**.
6. **Right-click** on the Reference Character and select **Hide All**. This hides all the entities that make up the reference character, including all mass objects, collision objects, graphical objects, joints and joint limits.
7. In the Node View, identify the node representing the graphical object **MayaChara_Mesh** that you created from the imported OBJ file. The graphical object name will match the corresponding OBJ file name. Click this node to display the graphical object of the mesh.
8. Select **Activate Simulation Character**. You will still be able to view the mesh, but it will no longer be selectable.



Unit Settings	
Angle	Degrees
Timeline Settings	
Frame Rate	60
Simulation Frame Rate	120
Simulation Steps To Run	10000
Time Magnification	1.0
Looping	☑
Bitmap Export	
Enabled	☐
Path	C:\Program Files\Natur…
Frame Rate	25
Frame Count	200
Horizontal Resolution	640
Vertical Resolution	480
Render View Only	☑
Simulation Settings	
Non-Spherical Mass Objects	☐
Gravity	
X	0.0
Y	-9.81
Z	0.0
Material Interactions	
Material Interaction	Bouncy
Restitution	1.0
Friction	100.0
Max Adhesive Force	0.0
Softness	0.0
Material Interaction	CustomA
Restitution	0.0
Friction	0.0
Max Adhesive Force	0.0
Softness	0.0
Material Interaction	CustomB
Restitution	0.0
Friction	0.0
Max Adhesive Force	0.0
Softness	0.0
Material Interaction	Default
Restitution	0.2
Friction	100.0
Max Adhesive Force	0.0
Softness	0.0
Material Interaction	Grippy
Restitution	0.2
Friction	100.0
Max Adhesive Force	0.0
Softness	0.0
Material Interaction	Icy

endorphin 2.5.Alpha
For news, support and downloads visit the NaturalMotion website at www.naturalmotion.com

9. You can now **move**, **rotate** and **scale** each of the simulation character collision objects to match the shape specified by the skin graphical object.

   You can also select individual mass objects and create **new** child collision objects. This is useful for filling out the mesh more accurately.

   You can use the **Mirror Tool** in Character Edit Mode to quickly copy changes from one side of the character to the other side.

   See the User Guide and tutorials for more detailed information on filling out simulation characters.

10. It is good practice to regularly **save** your changes when you are editing characters.

Step 5: Exporting animation from Maya into *endorphin*

In this step, you are going to **export** animation that you have created in Maya, and then **import** it into *endorphin*, using an FBX file to transfer the data.

To export animation from Maya:

1. Open the Maya scene **Resources\Tutorials\Tutorial 17– Working with Maya\MayaChara_RunJump.ma**.
2. Select **File > Export All**. Specify **MayaChara_RunJump** as the export filename and choose **FBX** file from the File Type list.
3. Click **Export** to create the FBX file.

To import animation into a new *endorphin* scene:

1. Select **File > New** to create a new scene.
2. Delete the default simulation character.
3. Select **Character > Add Character**. This displays the **Add Character** dialog.
4. Select the character that you created in Step 3 and Step 4. You may need to browse if you have saved that simulation-reference character to a different location. You do **not** need to turn on the Include Reference Character setting.
5. Select the character using the Timeline Editor, and then select **File > Import**. This displays the **Select File To Import** dialog.
6. Browse to select any FBX file that you created using Maya with the corresponding character. Click **Open** to display the **Import Options** dialog.
7. In the Import Option dialog, turn on the **Source Reference Character** setting.
8. Click the **Browse** button in the Auto Motion Transfer settings. Select the **MayaChara** reference character file you created for importing data in Step 2.
9. In the **General** settings, turn on the **Include parent transforms** setting.
10. In the node hierarchy tree, select the **Hips** node. This ensures that only motion on this joint—and its child joints—is imported.
11. Click **OK** to import the animation in the FBX file using dynamic motion transfer. Internally, *endorphin* imports this FBX animation onto the import reference character, and then uses dynamic motion transfer to map this motion onto the selected simulation character in your scene.

    You can import many different animation segments onto the **same** *endorphin* simulation character. Typically, you will create a separate **animation event** for each set of animation data keyframes, although you can also import the animation data directly as keyframes in the simulation character timeline.

Step 6: Exporting animation from *endorphin* into Maya

In this step, you are going to **export** animation that you have created in *endorphin*, and then **import** it into Maya, using an FBX file to transfer the data.

To export animation from *endorphin*:

1. Select the character that contains the animation data that you want to export.
2. Select **File > Export**. This displays the **Select File To Export** dialog.
3. Set the **Save As** type to FBX Files. You can enter a new name, or choose an existing FBX file to replace it. Click **Save** to display the **Export Options** dialog.
4. In the **Export Options** dialog, turn on the **Target Reference Character** setting.
5. Click the **Browse** button in the Auto Motion Transfer settings. Select the **MayaChara** reference character file you created for importing data in Step 2.
6. Click **OK** to export the animation to the FBX file using dynamic motion transfer. Internally, *endorphin* uses dynamic motion transfer to map the character motion onto the specified export reference character, and then exports this mapped motion.

To import animation into Maya:

1. In Maya, open the **MayaChara_RigidBind.ma** scene that you want to import animation into.
2. Select **File > Import** and browse for the FBX file that you saved from *endorphin*.
3. In the Import Dialog, turn on **Exclusive Merge** and **Pre-Normalize Weights**. Click **OK.**
4. Press **Play** in Maya to see the animation applied to the character. You can now modify this animation like any other Maya animation.

Conclusion

In this tutorial, we have configured the Maya animation pipeline. We have then imported animation data from Maya into *endorphin*, and exported animation data back from *endorphin* into Maya.

Tutorial 18

Working with the Maya Control Panel

In this tutorial, we will introduce you to the basic workflow for working with the *endorphin* Control Panel plugin for Maya. This plugin is available for Maya 7.0 and Maya 8.0, and is designed to improve the pipeline for exchanging animation data between Maya and *endorphin*.

Step 1: Adding an *endorphin* character to your Maya scene

In this step we will open a Maya scene, and add an *endorphin* character to this scene.

To add an *endorphin* character to a Maya scene:

1. Launch Maya, and open the **ExampleIKRig.mb** file. This file contains a very simple example of a typical IK keyframing rig.
2. Click the **endorphin Control Panel** button on the **endorphin** shelf. This launches the ***endorphin* Control Panel editor**.
3. On the **endorphin Character Editor** page, enter a proposed character name for your new character.
4. Click the **New Character** button to create a new *endorphin* character in the scene. A new tab will appear in the Control Panel notebook. The name of this tab will correspond to your character name.

Step 2: Reshaping the *endorphin* character

In this step we will reshape the *endorphin* character to match a Maya custom skeleton.

To reshape the *endorphin* character:

1. Reshaping the *endorphin* character requires **snapping** it to the custom skeleton. Turn on the **Snap To Points** editor setting on the Maya main toolbar.
2. Starting with the *endorphin* **root** joint, snap the *endorphin* joints—contained in the blue skeleton—to their corresponding joints in the Maya custom skeleton. In some cases, there is no one-to-one correspondence between the *endorphin* and Maya joints. In these cases, manually position the *endorphin* joints partway between the corresponding Maya joints.
3. When you have completed the reshaping process, click the **Save Character As** button. In the file browser, enter a filename that matches your character name, and click **Save**. This saves a copy of the *endorphin* character that contains your reshaped joint positions and orientations.

Step 3: Creating import connections

In this step we will create **import connections** that connect the *endorphin* character to the Maya character. These connections allow the *endorphin* import character to drive the Maya character.

To load the Maya character control rig root node:

1. Click on the notebook tab that corresponds to the name of the new *endorphin* character. This displays the **character editor notebook** for that character. The character editor notebook includes the Create Import Connections editor, the Create Export Connections editor, and the Exchange Animation Data editor.
2. Use the Maya Outliner or Hypergraph to select the topmost node of Maya control rig—the **Master** node.
3. In the **Create Import Connections** editor, click the **<< Load** button. The Load button loads all the child nodes of the selected Master node into the list of Maya character control rig nodes.

To create an import connection between the characters:

1. We now need to create import connections between the *endorphin* skeleton and the control rig nodes.

   In the diagram of the *endorphin* character, click on the **root** button. This is the button located at the centre of the character. The button will be displayed **yellow** to indicate that it is selected. The corresponding joint is selected in the Maya viewport. Also, the corresponding joint is selected in the list of *endorphin* character nodes below the diagram.

2. In the Maya viewport, select the **box graphic** that contains the root of the Maya custom character. You should find that the corresponding node is selected in the Maya character control rig nodes list.

3. With an *endorphin* node selected, and a Maya node selected, click the **Create Import Connection** button to create a connection between these nodes. The selected diagram button will now be displayed **red** to indicate that it is a connected node. In addition, the connected nodes are displayed as pairs of nodes in the *endorphin* and Maya node lists. With the root nodes connected, any animation applied to the *endorphin* character root will now drive the Maya character root.

To create a look-at import connection between the characters:

1. A common feature of Maya control rigs is the use of IK chains to define the motion of arms and legs. By connecting the *endorphin* **look-at helper nodes** to the IK pole vector nodes of the Maya custom character, you can use endorphin animation to drive the IK chains making up the Maya custom character, which then drive the underlying Maya skeleton in a manner which is similar to the original endorphin motion.

   The import character look-at helpers are displayed alongside the elbows and knees in the connection diagram. If you are not sure which button is associated with which joint, hover over the button to display its corresponding ToolTip. Alternatively, click the button to select its corresponding node in the node list underneath the diagram.

2. Click the **LeftKneeJointLookAtHelper** button.

3. In the Maya viewport, select the left leg look-at controller, which is represented by a **blue arrow graphic**.

4. Click the Create Import Connection button to create a connection between these nodes.

To fully connect the *endorphin* and Maya characters:

1. Once you are comfortable with the process of creating import connections, continue creating import connections between the *endorphin* and Maya characters. If you make an error, select either of the nodes that are connected, and click the **Delete Import Connection** button to delete the connection. Alternatively, click the **Delete All Import Connections** button to delete all of the import connections.
2. Each time you create a connection, a corresponding connection locator graphic is created in the Maya viewport. To toggle the visibility of these locators, select **Display > Display Connection Locators** menu item.
3. By default, the list of Maya control rig nodes displays only the node name itself. If you want to view entire node names—qualified by their full parenting path—select the **Display > List Maya Full Node Paths** item.

   Similarly, by default the lists of *endorphin* and Maya nodes display the full set of nodes. If you want to view only the nodes involved in import connections, select the **Display > List Connected Nodes Only**.

   Continue creating import connections until you have the following node pairs connected. Note that none of the endorphin end joints appear in the *endorphin* node list, or in the *endorphin* node diagram. These joints never contain animation data, and so never need to be connected to your Maya character. Also note that not **every** *endorphin* node requires connecting to a corresponding Maya character:

***endorphin* node**	**Maya node**
Root	HipsCTRL
LowerSpineJoint	Spine1CTRL
MidSpineJoint	Spine2CTRL
UpperSpineJoint	Spine3CTRL
LowerNeckJoint	NeckCTRL
UpperNeckJoint	HeadCTRL
LeftClavicleJoint	LeftShoulderCTRL
LeftElbowJointLookAtHelper	LeftElbowLookat1
LeftWristJoint	LeftHandCTRL
LeftFingersJoint	LeftFingersCTRL

LeftKneeJointLookAtHelper	LeftKneeLookat1
LeftAnkleJoint	LeftFootCTRL
RightClavicleJoint	RightShoulderCTRL
RightElbowJointLookAtHelper	RightElbowLookat1
RightWristJoint	RightHandCTRL
RightFingersJoint	RightFingersCTRL
RightKneeJointLookAtHelper	RightKneeLookat1
RightAnkleJoint	RightFootCTRL

Step 4: Creating export connections

In this step we will create **export connections** that connect the *endorphin* character to the Maya character. These connections allow Maya character to drive the *endorphin* export character.

To load the Maya character skeletal rig root node:

1. Click on the notebook tab that corresponds to the name of the new *endorphin* character. This displays the **character editor notebook** for that character. The character editor notebook includes the Create Import Connections editor, the Create Export Connections editor, and the Exchange Animation Data editor.
2. In the **Create Export Connections** editor, the skeletal node has already been loaded, and we do not need to change this setting. In fact, now that we have reshaped the endorphin character to match the Maya skeleton, we can very quickly create a set of export connections automatically, using a simple proximity test.
3. Click the **Quick Create Export Connections** button to create the export connections based on proximity, and click OK in the confirmation dialog. The Control Panel compares the position of each *endorphin* joint to each Maya joint, and connects pairs of joints that are located within a tight distance threshold from each other.
4. In this example, we are satisfied with the export connections created by this command. However, you are able to create and delete individual export connections using the Create Export Connection and Delete Export Connection commands, just as for import connections.

Step 5: Creating sample *endorphin* animation

In this step we will launch *endorphin*, and create some sample *endorphin* animation that uses the *endorphin* character that we created in Maya in Step 2.

To create sample *endorphin* animation:

1. Launch *endorphin*, and create a new scene. Delete the default **Character01** character.
2. Click the **Add Character** button, and browse to the folder in which the *endorphin* character created in Step 2 was saved.
3. Select the character from the character list, and click **OK** to load this character into your scene.

   At this stage, the *endorphin* character will appear somewhat unusual. This is because the *endorphin* Control Panel for Maya plugin does not yet support the editing of mass and collision object positions and orientations. Future releases of the plugin may support additional character-editing functionality; for the moment, the *endorphin* **Character Edit Mode** must be used to complete the character-editing process.

   In the current tutorial—in which we are generating some sample animation to demonstrate the animation pipeline—we can ignore the incomplete editing state of the character. Of course, when editing characters to create actual animation, you will need to use Character Edit Mode to adjust the mass and collision objects and joint limits to ensure that they match the physical dimensions of your Maya character.
4. Right-click on the character timeline and select **Create Behaviour Event** to create a new behaviour.
5. With the event marker selected, use the Property Editor to set it Name property to **Stagger 3**. Also, set the Start Frame to 0, and the End Frame to 200.
6. Right-click on the character timeline and select **Create Force Event** to create a new force event. Set the Frame of the fore to 0.
7. Simulate the scene to generate some sample animation.
8. Select the character.
9. Select **File > Export**, and save the animation to an FBX file.
10. Close *endorphin*.

Step 6: Importing *endorphin* animation into Maya

In this step we will import the sample animation created in Step 5 into our Maya scene. This motion will be applied to the *endorphin* import character. The import connections created in Step 3 will drive the corresponding Maya character using this motion.

To import *endorphin* animation into Maya:

1. In Maya, open the *endorphin* Control Panel, and browse to the **Exchange Animation Data** tab associated with the *endorphin* character in the scene. There should be only one endorphin character in this scene.
2. Click the **Import Animation** button, and browse to the FBX file containing the animation generated in Step 5.
3. Click **Import** to import this motion onto the *endorphin* import character.
4. Use the Maya Time Slider to scrub through the animation. You should find that the *endorphin* import character is correctly animated by the animation. Importantly, however, you should also find that your Maya control rig is now driven by the *endorphin* motion as well. Note that the Maya control rig does not have any keyframed motion itself—instead, it is constrained to the keyframed motion of the *endorphin* import character.

Step 7: Modifying the imported *endorphin* animation

In this step we will modify the motion of the Maya control rig. You can modify the motion in a number of ways. In particular, on a node-by-bode basis you **add** to the *endorphin* motion, **replace** the *endorphin* motion, or create a **blend** between your motion and the *endorphin* motion.

To animate over the *endorphin* motion:

1. We are driving the Maya control rig using the import connection constraints created in Step 3. However, we can now **animate** the control rig **over** this motion. That is, we can use the *endorphin* sample animation as a **baseline** on top of which we animate. In this example, we want to change the motion of the left hand of the character such that it reaches backwards as though reaching for an object.

2. Select the Maya **LeftHandCTRL** node, which is represented as a blue box graphic.
3. Move the time slider to frame **0**, and set a key.
4. Move the time slider to frame **25**, and set a key.
5. Move the time slider to frame **10**, and adjust the position of this control node. Move the hand back so that the character is reaching behind itself, then set a key.
6. Scrub through the timeline to see the new Maya animation. The Maya character is still driven by the *endorphin* import character; on top of this motion we have added some additional left arm movement. In this manner you can modify the entire *endorphin* motion to stylise it as required by your target scene.

To replace the *endorphin* motion with animation:

1. In the previous steps, you **added** to the motion transferred from *endorphin* to Maya via the import connections. However, you can also reduce the strength of the constraints that transfer this motion. By reducing the strength of these constraints to zero, you can entirely **replace** the *endorphin* motion with your own motion. By choosing an intermediate setting for these constraints, you can **blend** between the *endorphin* motion and your own motion.

   There are separate position and orientation blend strength settings for each import connection. Blend strengths may vary from a maximum of 10.0 to a minimum of 0.0.

   In this example, we would like to reduce the *endorphin* contribution of motion to zero for a few frames, in order to completely replace it with our own motion.
2. With the hand control object still selected, in the Maya timeline, **right-click** and **copy** the key at frame **10**.
3. Move the time slider to frame **20** and paste the key.
4. Move the time slider back to frame **10**.
5. On the **Exchange Animation Data** page of the *endorphin* Control Panel, locate the position and orientation blend strength controls for the left hand of your Maya character. It will be labelled **LeftHandCTL** / **endorphin LeftFingersJoint**. These are the Maya and *endorphin* nodes, respectively, that are connected by the import connection.

6. Click the large **K** button to create keys for both the translation and rotation blend strengths. At this stage, both of these strengths should be at their **maximum** (10.0).
7. Move the time slider to frame **11**.
8. Set the translation and rotation blend strengths to their **minimum** (0.0).
9. Click the K button to create new blend strength keys for both translation and rotation.
10. Move the time slider to frame **20**.
11. Click the K button to create new blend strength keys for both translation and rotation.
12. Move the time slider to frame **25**.
13. Set the translation and rotation blend strengths to their **maximum** (10.0).
14. Click the K button to create new blend strength keys for both translation and rotation.
15. Scrub through the animation to see how changing the translation and rotation blend strengths affects the transfer of motion from the *endorphin* character to the Maya character.
    - Between frames 0 and 10, the blend strengths are at their **maximum**, and the *endorphin* left hand motion is entirely transferred to the Maya character.
    - Between frames 11 and 20, the blend strengths are **zero**, and none of this motion is transferred. Any left hand motion is entirely generated by your own keys.
    - Between frames 21 and 25, the blend strengths **gradually increase** from zero back to their maximum. Over this range, more and more of the *endorphin* left hand motion is applied to the Maya character left hand control.
    - From frame 25 onwards, the blend strengths are at their **maximum**, and *endorphin* is once again fully driving the left hand control.

In this way, you can not only animate over the top of the *endorphin* motion, you can also replace the *endorphin* motion altogether if required.

Conclusion

In this tutorial, you have used the *endorphin* Control Panel to create, reshape and save endorphin characters. You have also created import and export connections to associate *endorphin* characters with corresponding Maya characters.

You have then created animation in *endorphin*, and imported it onto *endorphin* characters in Maya. This motion has then driven connected Maya characters via the import connections that you created.

You then experimented with animating the Maya control rig over the *endorphin* motion, and finally, by adjusting the relative blend strengths over time, replaced the *endorphin* motion with your own motion for certain periods. In this way you have taken motion generated by *endorphin*, and customised it to the needs of your Maya scene.