UserGuide

Contents

Chapter 1

Getting Started

Welcome to *endorphin*

What is *endorphin*?

NaturalMotion's *endorphin* is the industry's first dynamic motion synthesis software system and is probably unlike anything you have seen before. It uses revolutionary artificial intelligence and biomechanics techniques to precisely simulate the human body both physically and behaviourally.

At the heart of *endorphin* are its adaptive **behaviours**, which—unlike animation data—are completely interactive. With adaptive behaviours, three dimensional characters essentially animate themselves. Move two football players close to each other and one will automatically tackle the other one, realistically trying to grab hold of its legs and bringing it down. Or not. It really is up to you.

endorphin allows animators to direct scenes in real time in a way they have never been able to do before. You can change parameters or change behaviours and see the results instantly. It is software that will not only save you time, it will save you money. *endorphin* is not a crowd simulator or a replacement for current 3D packages. It represents a new way to animate—and is the way that all character animation will be created in the future.

Pepsi Max **Glue Boy** advertisement. Post-production using *endorphin* by The Mill, London. Courtesy Jordi Bares and The Mill.

Key features of *endorphin*

What are some of the key features of *endorphin*?

Some of the key features of the *endorphin* dynamic motion synthesis system include:

- **High-fidelity characters:** *endorphin* uses high-fidelity biomechanical models of the human body. Character dimensions and joint limits are based on anthropometric data sources. The biomechanical properties of characters are enhanced by Oxford University research.
- **Intuitive character editing:** *endorphin* includes powerful character editing tools that allow you to change the shape, dimensions and mass of the *endorphin* character to match your custom characters.
- **Adaptive behaviours:** *endorphin* ships with a Behaviour Library containing a wide selection of adaptive character behaviours generated using artificial intelligence techniques. Characters can sense and react to their simulated virtual world. The Behaviour Library is based on extensive surveys of the demands of game developers and post-production studios. Plus, with the expandable Behaviour Library, new behaviours can be incorporated into *endorphin* as they become available. *endorphin* behaviours are available with intuitive user-level parameters, such as Urgency, Strength and Speed, so you can easily fine-tune your animation.
- **High-quality dynamic virtual world:** *endorphin* uses a robust, fast rigid-body dynamics engine to generate very accurate physical simulated interactions between characters and their environment. Characters feature realistic collision surfaces and mass properties, and have full support for additional attached objects, such as armour and weapons. Controlled detachment of body parts and props is also easy to use.
- **Multiple character scenes:** *endorphin* lets you create scenes with multiple, interacting real-time characters, all with independent behaviours and poses.
- **Dynamic motion transfer:** *endorphin* proprietary motion mapping techniques let you import and map your animation data onto characters, generate new animation for these characters, and then export the animation out back onto your characters.
- **Pipeline integration:** *endorphin* can import and export character hierarchies and animation data in a variety of industry-standard file formats. Supported file formats include FBX, XSI and BVH file formats, which are used to create pipelines between *endorphin* and animation tools including Maya, 3dsMax, SOFTTIMAGE|XSI, MotionBuilder, Lightwave and Poser. Other supported formats include the Vicon motion capture format and the Acclaim format.

- **Animated cameras:** *endorphin* lets you import animated camera paths and supports live-action video back plane using AVI files. Together, these features let you visualise your characters interact with live action footage in real time.
- **Active character posing:** *endorphin* lets can define multiple poses and use them as dynamic keyframes—characters will try and achieve their active poses in a believable and fluid manner.
- **Interactive strobing:** *endorphin* strobe preview allows you visualise the end results of iterative changes instantly, using a presimulation that is faster than real time.

What's new in *endorphin* 2.6

Application

- **Network licensing** now allows you to run multiple instances of *endorphin* from a single, server based hardware lock.
- **User guide** has been significantly enhanced.

Behaviours

- The **Hold** behaviour is new. It allows you to automatically trigger constraint events in response to mass object proximity events. This behaviour reduces the need to manually adjust the timing of constraints while a scene is under development, and makes scenes more robust against changes.
- The **Balance With Props** behaviour is new. It extends the **Balance 2.5** behaviour to take into account the mass of any props that are associated with the balancing character.
- Many behaviours in the Behaviour Library now have **helper graphics**. These graphics allow you to interpret motion generated by a behaviour.

Editing scenes

- When you create new mass objects in scenes, they are now automatically created with an explicit **child collision object** of the same shape. These default child collision objects may be edited in the same manner as any other collision object. Previously, mass objects contained an embedded collision object that was unable to be deleted. Added in *endorphin* 2.6.1.

Character Edit Mode

- Connections between mass objects and rig nodes are now **automatically** created when the Move tool is used to snap simulation character joints to reference character

joints. Previously, all connections between mass objects and rig nodes had to be specified manually in the Character Rig Setup dialog.

- When simulation and reference character joints have been **snapped** together, their positions are now **locked** and not affected by use of the Move, Rotate or Scale tools on joints higher up the joint hierarchy. Previously, snapped joint positions were not locked, and so could be affected by when editing joints higher up the joint hierarchy.
- The **root** joint of the simulation character can be snapped to the root joint of the reference character. Previously, snapping was not available for root joints of characters.
- Characters and joints can now both be **scaled** using the Scale tool.
- When editing simulation-reference pairs in Character Edit Mode, you can now more readily toggle the active character by selecting **Edit > Toggle Active Character**. The keyboard shortcut is **Ctrl+Tab**. Added in *endorphin* 2.6.1.

Character Rig Setup dialog

- This dialog is now a tool window, and has been renamed the **Motion Transfer Editor**. The keyboard shortcut to display this editor is now **M**. This editor is **modeless**, which means you can continue working with *endorphin* while this editor is being displayed. Previously, this dialog was modal, which meant that you could not work on endorphin while it was being shown.
- You can now edit **multiple** rig groups and multiple rig nodes at the same time. Previously, you could only select a single rig group and a single rig node at any one time.
- You can now create new rig groups by **copying** existing rig groups.

Other tool windows

- All tool windows (Property Editor, Node View, Timeline Editor, Motion Transfer Editor, Scene Notes, Output) now have a **titlebar** displayed even when docked. This titlebar allows you to quickly identify the type of tool window, and makes it easier to float the tool window—by double-clicking or dragging the titlebar—and to close the window.
- The Console View has been renamed as the **Output**.

Property Editor

- The Property Editor now displays the properties for **multiple** selected objects in a new way. It now displays the properties that are **common** to the selected objects. Previously, it displayed a separate property sheet for each selected object. This change makes it a lot easier to quickly edit the same property on multiple objects.

Node View

- You can now select multiple objects using the **Node View**. Previously, only one object at a time could be selected.
- Adding and removing objects from the scene no longer causes the Node View to return to its fully-collapsed state. It now maintains the individual expanded and collapsed settings for each node in the Node View. This makes the Node View easier to use, particularly in Character Edit Mode. Added in *endorphin* 2.6.1.

Scene Notes

- The **Scene Notes** dialog is now a modeless tool window, which means you can continue to work with *endorphin* while it is open. It also has improved text formatting capabilities.

Selection

- Selecting objects in viewports is now **cyclic**. When you click in the viewport, the object closest to you is selected, as before. However, if you click at the same screen position, the next farthest object is selected, and so on, in a cyclic manner. Also, viewport selection is more precise than in previous versions of *endorphin*.
- When mass objects and collision objects are **pasted**, you can now choose collision objects, graphical objects and joints when pasting the objects. Previously, you could only select mass objects.

Menus

- The main menu has been slightly reorganized.
- The context menus for the Timeline Editor, Node View and viewports have been updated. You can now show and hide characters using context menu commands using the Timeline Editor and viewport context menus. You can also freeze and unfreeze graphical object selection using the Node View context menu.

Data import and export

- You can now import **animated joint lengths** when importing FBX files. **Note** that there are some graphics visualization issues with this functionality that will be addressed in the next release of *endorphin*.
- The **Export Animation Into Scene** functionality in the Export Options dialog has been replaced with the **Create File From Template** functionality. These functions differ in that previously, you modified the selected file. The new approach is to always create a new file by merging with the selected file.

Keyboard shortcuts

- The keyboard shortcut for **Toggle Camera Tracking** is **C**. Previously it was **Ctrl+T**.
- The keyboard shortcut for **Toggle Strobing** is **Ctrl+Shift+S**. Previously it was **Alt+S**.
- The keyboard shortcut for **Full Screen** is **Ctrl+Shift+F**. Previously it was **Alt+F**.
- The keyboard shortcut for **Motion Transfer Editor** is **M**. Previously it was **C** (for Character Rig Setup dialog).

Scripts

- The Maya MEL script **endorphin_Exporter.mel** has been added to improve the *endorphin*-Maya pipeline. This script replaces the prop character exporter script and camera exporter script that were shipped in previous versions of *endorphin*, and allows you to create correct jointed hierarchies—including mass and collision objects—directly in Maya. Properties such as density and material are also supported as Maya attributes. Added in *endorphin* 2.6.1.

Advantages of using *endorphin*

How can *endorphin* help you?

Some of the unique benefits of using the *endorphin* dynamic motion synthesis system include:

- **Rapid turnaround:** *endorphin* is orders of magnitude faster than any other software on the market, so you can produce animation hundreds of times more rapidly than with conventional methods.
- **Increased creativity:** *endorphin* generated animation in real time, not only is your productivity expanded, but your creativity is boosted as well. You can make changes later and later in the production process, and collaborative working becomes possible.
- **Interactive visualisation:** *endorphin* strobe previews gives you faster than real-time pre-simulation visualisation, allowing you to see the end results of any changes you make instantly.
- **Behavioural realism:** *endorphin* adaptive behaviours generate nuanced human animation that is as good—and often better—than animation produced by keyframing or motion capture.
- **Pipeline integration:** *endorphin* fits easily into your existing animation pipeline. You can import your characters, generate animation for your characters, then export clean animation data for rendering or into a games engine.

endorphin Learning Edition

***endorphin* Learning Edition** is a free, time-unlimited version of *endorphin*. *endorphin* Learning Edition includes the full range of *endorphin* functionality, with the exception of animation data export, which is restricted to AVI export.

To download *endorphin* Learning Edition, visit **www.naturalmotion.com/pages/le.htm**.

File formats

endorphin Learning Edition uses a **different file format** to the full version of *endorphin*. Scene files, character files and pose files generated by *endorphin* Learning Edition cannot be loaded in the full version of *endorphin*. Similarly, scene files, character files and pose files generated by the full version of *endorphin* cannot be loaded in *endorphin* Learning Edition.

File extensions

To distinguish *endorphin* and *endorphin* Learning Edition files, **different file extensions** are used by the two systems. This guide uses the *endorphin* file extensions throughout. The different file extensions do not affect the functionality in any way. Readers who are using *endorphin* Learning Edition should note that where the where this guide refers to a specific file extension, they should replace it with the corresponding *endorphin* Learning Edition file extension.

- **Scene files:** *endorphin* uses **.ens**, whereas *endorphin* Learning Edition uses **.els**.
- **Character files:** *endorphin* uses **.nmc**, whereas *endorphin* Learning Edition uses **.elc**.
- **Pose files:** *endorphin* uses **.nma**, whereas *endorphin* Learning Edition uses **.elp**.

Watermarks

You can distinguish *endorphin* and *endorphin* Learning Edition viewports by the presence of the Learning Edition **watermark** in the lower-right corner of *endorphin* Learning Edition viewports. The watermark is a reminder that you cannot use the *endorphin* Learning Edition for commercial use.

Automatic software upgrade notifications

Each time you launch *endorphin* Learning Edition, *endorphin* checks to see if a newer version of *endorphin* Learning Edition is available for download. If a newer version exists, an **Upgrade** dialog appears to inform you that a new version is available. Note that this function is only available if you have an internet connection when launching *endorphin* Learning Edition.

Th automatic software upgrade notification feature is only available with *endorphin* Learning Edition—if you are a licensed user of *endorphin* or *endorphin* Educational Edition, you will be informed of newer versions of *endorphin* directly, either by your reseller or by **NaturalMotion**.

endorphin Educational Edition

***endorphin* Educational Edition** is a reduced-cost, time-unlimited version of *endorphin* for educational facilities. *endorphin* Educational Edition includes the full range of *endorphin* functionality, including animation data export.

File formats and extensions

endorphin Educational Edition uses the **same** file formats as the full version of *endorphin*. Scene files, character files and pose files generated by *endorphin* Educational Edition can all be loaded in the full version of *endorphin*. Similarly, scene files, character files and pose files generated by the full version of *endorphin* can be loaded in *endorphin* Educational Edition.

endorphin Educational Edition uses the same file extensions for scene files, character files and pose files as the full version of endorphin.

Watermarks

You can distinguish *endorphin* and *endorphin* Educational Edition viewports by the presence of the Educational Edition **watermark** in the lower-right corner of *endorphin* Educational Edition viewports. The watermark is a reminder that you cannot use the *endorphin* Educational Edition for commercial use.

Version history

endorphin was released in August 2003. Since that time, there have been continual enhancements. Most major new interface and functionality changes occur with major releases. Minor, or point releases, often follow a major release in order to further refine or enhance these major changes.

endorphin 1.1

- *endorphin* 1.1.0 August 2003
- *endorphin* 1.1.1 September 2003

endorphin 1.2

- *endorphin* 1.2.0 November 2003

endorphin 1.3

- *endorphin* 1.3.0 February 2004

endorphin 1.5

Introduced the User Guide.

- *endorphin* 1.5.0 May 2004

endorphin 1.6

Introduced Character Edit Mode; environment materials; XSI support; Vicon support; animated cameras and video planes; more context menus.

- *endorphin* 1.6.0 January 2005
- *endorphin* 1.6.1 February 2005

endorphin 2.0

Introduced Auto Motion Transfer during import and export; animation events; more flexible Timeline Editor; new colour scheme; introduced Learning Editions

- *endorphin* 2.0.0 March 2005
- *endorphin* 2.0.1 April 2005
- *endorphin* 2.0.2 May 2005

endorphin 2.5

Introduced active animation events; introduced transition events; added Mirror tools to Character Edit Mode; added Character Rig Setup dialog.

- *endorphin* 2.5.0 October 2005
- *endorphin* 2.5.1 November 2005
- *endorphin* 2.5.2 December 2005
- *endorphin* 2.5.3 March 2006

endorphin 2.6

Introduced Motion Transfer Editor; enhanced Character Edit Mode; introduced behaviour visualisation; improved User Guide.

- *endorphin* 2.6.0 May 2006
- *endorphin* 2.6.1 June 2006

Installing *endorphin*

To install *endorphin*, perform the following steps:

1. Place the **enclosed CD** into the CD-ROM drive of your computer. If you are installing from a download, **double-click** on the downloaded installation file.

The installation program should start automatically. If it doesn't, run **endorphinSetup.exe**, located in the root directory of the CD.

2. Follow the onscreen instructions to specify an installation folder. The default location is **C:\Program Files\NaturalMotion**. You can install multiple versions of *endorphin* on the same computer.
3. Follow the onscreen instructions to specify the name of a folder that will appear in the Windows **Start Menu**. The default location is under **NaturalMotion**.
4. Follow the onscreen instructions to specify whether to create a **desktop icon**, and whether to create a **QuickLaunch** icon on the Windows Taskbar.

Note You do **not** need to uninstall *endorphin* before installing a newer version. Also, you do not need to uninstall one edition—such as the Learning Edition—before installing another edition. However, keep in mind that the various *endorphin* file extensions will be associated with the most recently installation of *endorphin*.

To uninstall *endorphin*, perform the following steps:

1. Close all instances of *endorphin*.
2. Select **Start > Control Panel** from the Windows Taskbar.
3. Double-click on the **Add or Remove Programs** item. This displays the **Add or Remove Programs** dialog.
4. Locate the installation of **NaturalMotion endorphin** that you want to uninstall.
5. Click the **Remove** button. This runs the uninstaller utility **unins0000.exe**, which is located in your installation folder. Alternatively, you can double-click on this utility directly in the Windows File Explorer.
6. Click the **Yes** button in the Uninstall dialog. *endorphin* will be completely uninstalled from your system.

To license *endorphin* or *endorphin* Educational Edition, perform the following steps:

- Insert the enclosed **hardware lock** into a free USB port on your target machine. This USB port is not required to be powered.
- If you have a **network license** of *endorphin*, insert the hardware lock into a free USB port of the **network** machine. You will then be able to run instances of *endorphin* on client machines without the need for a hardware lock on the client machine. Keep in mind that each network license has a **maximum** allowed number of concurrent users. This maximum number varies and depends upon on your license agreement.

To license *endorphin* Learning Edition, perform the following steps:

You do **not** need to have a hardware lock to run *endorphin* Learning Edition. However, you will need to register with NaturalMotion in order to receive a free Product Key. This requires access to the internet.

1. The first time you run *endorphin*, the **License Key** dialog appears.
2. Click the **Register Now** link in the License Key dialog.
3. The *endorphin* **User Community Forum** website will be displayed in a new browser.
4. Click the **Profile Page** link to create a user profile. You will need to answer a short set of Survey Questions. These questions are designed to help NaturalMotion improve your user experience in future versions of *endorphin*. When you have completed the Survey Questions, an *endorphin* Product Key will be displayed on this page.

5. Enter your **Email Address** and the **Product Key** in the corresponding fields in the License Key dialog, and click **OK**. This will register your copy of the *endorphin* Learning Edition. You will now be able to run *endorphin* Learning Edition. You will **not** be asked to supply this Product Key again.

Hardware lock expiration date

If you are using a licensed version of *endorphin*, *endorphin* Educational Edition or *endorphin* Learning Edition, you can use *endorphin* for an unlimited period of time.

However, if you are using an **evaluation license** of *endorphin* or *endorphin* Educational Edition—or if you are using a rental licensing scheme such as *endorphin* **Rush**—then your license of *endorphin* will usually have an expiration date after which it will no longer operate.

You can find the expiration date of your license by selecting **Help > About endorphin…** to display the **About NaturalMotion *endorphin*** dialog. This dialog displays your hardware lock expiration date. If this textbox is empty, you are using a time-unlimited license.

Network licenses

If you are using a network license of *endorphin* or *endorphin* Educational Edition, your license specifies a **maximum number** of different users that can run *endorphin* at the same time. This number will depend upon your licensing arrangements with NaturalMotion. Keep in mind that a single user can run an unlimited number of instances of *endorphin*—the only restriction is in the number of **different** users.

endorphin regularly checks the network to ensure that the maximum number of users has not been exceeded. Once the maximum number of unique users has been reached, no additional users will be able to launch *endorphin*. If you have reached the maximum number of users, and one or more users end all their *endorphin* sessions, you may notice a one to two minute delay before different new users are able to launch *endorphin*.

System requirements

Hardware requirements

- **Processor:** An Intel Pentium or AMD Athlon processor with a clock speed of 1.7GHz or higher. **Intel:** Pentium D, Pentium M and Pentium 4. **AMD:** Athlon, Duron, Opteron, Athlon 64 and Athlon 64 X2.
- **Memory:** At least 512MB.
- **Graphics:** Any recent ATI or nVidia graphics card. **ATI:** Radeon Mobility, 7000, 8000, 9000 and X series; FireGL V or X Series. **nVidia:** GeForce 2, 3, 4, and FX 5 and 6 series; Quadro 4 and Quadro FX series; GeForce Go cards.

Software requirements

Operating system: Windows 2000 or Windows XP only. *endorphin* uses a number of system features that have been documented as functioning incorrectly on Windows 95, Window 98 and Windows ME. As a result, it highly likely that you will experience some instability if it is run under these operating systems. Also, as Microsoft no longer supports them, we have no opportunity to correct this issue and are, therefore, unable to support operating systems other than Windows 2000 and Windows XP.

- **Video drivers for desktop computers:** These are the current driver releases we support: ATI Radeon: Radeon unified drivers: Catalyst 5.8. ATI FireGL: FireGL unified drivers: 8.133.1.1. Both of these are available from: http://support.ati.com/. nVidia Quadro: 77.56 WHQL Certified Driver. nVidia GeForce: ForceWare 78.01 WHQL Certified Driver. Both of these are available from: http://www.nvidia.com/content/drivers/drivers.asp.
- **Video drivers for laptop computers:** In most cases, the above desktop drivers do not support video hardware in laptop computers. Where this occurs, it is usually necessary to visit the website of your laptop's manufacturer in order to download their current driver release.

Getting Started dialog

When *endorphin* is launched, the **Getting Started** dialog appears by default.

This dialog is useful for new *endorphin* users, and includes links to the following resources:

- **NaturalMotion** homepage.
- ***endorphin* User Community Forum** homepage.
- **Tutorial Videos** page on the NaturalMotion site. The tutorial videos are a collection of **AVI** files that take you through each of the tutorials in the User Guide. You can download these files individually. Typical file sizes are 10MB to 20MB. Keep in mind that the tutorial videos are produced **after** each new release of *endorphin*. This means that the set of tutorial videos may not always correspond to the very latest release of *endorphin*.
- **Sample Scenes** folder in your installation.
- **User Guide** in your installation.

You can prevent this dialog from reappearing each time you launch endorphin by turning off the **Show this dialog on startup** setting. You can access this dialog at any time by selecting **Help > endorphin Getting Started**.

Sample scenes

Getting started with *endorphin*

A good way to start exploring *endorphin* is to try loading some of the **sample scenes**. We have provided a number of sample scenes that illustrate various features. The sample scenes can be found in the **Resources\Scenes\Sample Scenes** folder.

We recommend just playing around with the settings and seeing what you get as an end result. Changing surface friction settings, for example, has a very obvious effect when it comes to executing a football tackle. Specifying zero gravity is another way to quickly change the outcome of a simulation.

Tutorials

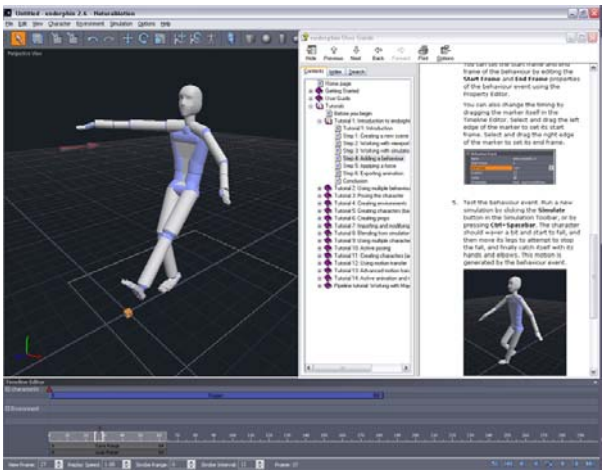
Learning more about *endorphin*

More than two hours of *endorphin* tutorial videos have been included on the installation CD to help you get to grips with the power, scope and ease of use of the software. These include:

- **Tutorial 1:** Introduction to *endorphin*
- **Tutorial 2:** Using multiple behaviours
- **Tutorial 3:** Posing characters
- **Tutorial 4:** Building environments
- **Tutorial 5:** Basic character creation

- **Tutorial 6:** Creating prop characters
- **Tutorial 7:** Importing and modifying existing animation data
- **Tutorial 8:** Blending from simulation to animation
- **Tutorial 9:** Multiple character scenes
- **Tutorial 10:** Active posing
- **Tutorial 11:** Advanced character creation
- **Tutorial 12:** Using dynamic motion transfer
- **Tutorial 13:** Advanced motion transfer techniques
- **Tutorial 14:** Active animation
- **Pipeline tutorial:** Working with Maya

The tutorial scenes can be found in the **Resources\Tutorials** folder.



User community forum

Get more out of *endorphin*

The *endorphin* **User Forum** is a site where you can discuss and exchange ideas about *endorphin*. You can find tutorials, sample scenes, tips and techniques, FAQs and more.

To visit the *endorphin* User Forum, visit **community.naturalmotion.com**.

Contacting us

For support and sales enquiries:

You can get in touch with us in the following ways:

- **Web:** Visit our support page at **www.naturalmotion.com/pages/support.htm**.
- **Email:** contact@naturalmotion.com.
- **Phone:** +44 1865 250 575.
- **Fax:** +44 1865 250 577.
- **Address:** NaturalMotion Limited, Beaver House, Hythe Bridge Street, Oxford OX1 2ET, United Kingdom.

Getting around this guide

Contents

This guide is made up of a set of **topic** pages. Topics are collected in **books**, and the topic books themselves are arranged into an overall **Contents** tree.

Use the Contents pane to browse to different topics. You can use the **UpArrow** and **DownArrow** keys to navigate up and down the Contents tree. You can also use the

LeftArrow and **RightArrow** keys to collapse and expand the topic book collections. When you have browsed to a specific topic, press **Enter** to display that topic.

Index

Each topic has a corresponding index term. The Index list displays an alphabetically sorted list of index terms. You can browse this list to display the corresponding topic.

Search

You can search for particular words or phrases using the Search textbox. Type in a search string, and then press **Enter** or click the **List Topics** button to display a list of the topics that contain the corresponding search phrase.

Back and Forward buttons

The help system keeps track of the topics that you have read. You can use the **Back** and **Forward** buttons to browse through the list of topics that you have read. The list of topics is cleared each time the help system is launched.

Previous and Next buttons

This guide is designed to be read sequentially. The sequence of topics in the Contents tree can be easily read sequentially by using the **Previous** and **Next** buttons.

Conventions used in this guide

Terminology

In this guide, you will often see instructions such as "Select **File > New Scene…**". This is a shorthand way to ask you to click on the **File** menu item in the Menu Bar, and then click on the **New Scene…** menu item.

Unless indicated, the term **click** means a **left mouse button** click, and the term **right-click** means a **right mouse button** click.

The terms **check**, **checking, checkbox** and **check item** refer to Windows controls and operations that involve toggling a setting between two states. In some regions, these controls and operations are more commonly referred to as **tick**, **ticking**, **tickbox** and **tick item**.

Operating system

Many of the graphics used in this guide were made using *endorphin* running on Windows XP. If you are running on a different version of Windows, you may notice subtle differences in the appearance of the menus and windows. These differences do not affect the performance of the software.

Keyboard options

Many *endorphin* commands have a corresponding keyboard shortcut, which can be customised. In this guide, whenever a keyboard shortcut is described, it is understood to be the default shortcut.

Colour options

Many aspects of *endorphin* user interface, such as colours, which you can customise. In this guide, whenever a colour is described, it is understood to be the default colour

Chapter 2

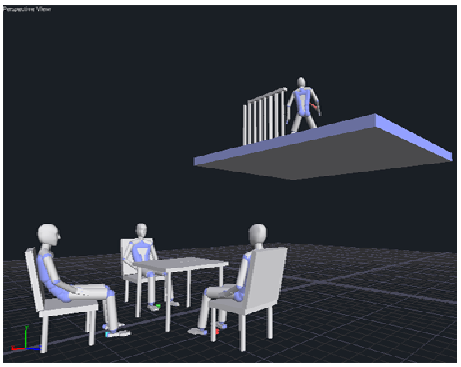
Basic Concepts

What is the *endorphin* virtual world?

The *endorphin* **virtual world** is a three-dimensional space that is subject to **physical processes**. Objects in the virtual world can have mass, velocity and momentum, and can be affected by gravity. Surfaces have material properties, which can affect collisions.

You can control which objects in the virtual world are subject to physical processes. You can specify which objects are **animated** and which objects are **simulated**. The motion of animated objects is controlled by keyframes, whereas the motion of simulated objects depends upon physical processes such as forces and collisions.

You can populate the virtual world with **characters**. Characters are also subject to physical processes, and can also be keyframed. However, characters can also interact with their environment—and with other characters—in an autonomous and intelligent way.



Coordinate system

The virtual world coordinate system uses global Cartesian coordinates centred on the world **origin**. The *endorphin* coordinate system assumes that the global **Y direction** is the **vertical direction**.

You may need to transfer animation data to or from other systems that use a different coordinate system. In particular, some systems assume that the Z direction is the vertical direction. You can specify this setting when you import or export data.

The global coordinate system axis is displayed in the **lower-left** corner of the viewport. The global **X, Y** and **Z** axes are represented using **red**, **green** and **blue** lines.

Units

By default, *endorphin* uses the **metric** system. For example, positions and distances are displayed in **metres**; velocities are displayed in **metres per second**; accelerations are displayed in **meters per second squared**; and densities are displayed in **tons per cubic metre**.

Angular positions are displayed in **degrees**. However, you can change this setting by changing the **Angle** property in the **Unit Settings** section of the Property Editor. Angular positions can also be displayed in radians and revolutions.

Keep in mind that many properties use dimensionless scale factors values rather than absolute values. For example, the strength of a force event is presented as a value between 0.001 and 50.0. However, this is **not** a force value in Newtons. Rather, it is a scale factor used in the application of an acceleration..

Gravity

The virtual world has a **gravitational** force field. This force field affects every mass object in the virtual world. You can change the **magnitude** and **direction** of the gravity field. You can also turn gravity off if required.

Gravity is turned on by default, and is directed vertically downwards with a magnitude corresponding to Earth surface gravity. This generates an acceleration in mass objects of 9.81 metres per second per second.

Materials

The virtual world has a number of predefined **materials** and **material interactions**. Each collision object is assigned one of the material types. When two collision objects touch during a simulation, the nature of the interaction depends upon the materials of each of the collision objects.

Materials

endorphin has a number of materials predefined. These are: Default, Ice, Rubber, Skin, Sponge, Steel, Tarmac and Velcro. Materials themselves do not have any parameters.

Material interactions

endorphin has a number of material interactions predefined. These are Default, Bouncy, Grippy, Icy, Soft, Sticky, CustomA and CustomB. You can change the restitution, friction, softness and maximum adhesive force for each material interaction.

Restitution describes the **elasticity** of the collision between objects. Restitution is always positive. A value of 0.0 indicates a perfectly **inelastic** collision, while a value of 1.0 indicates a perfectly **elastic** collision. A highly elastic collision conserves most of the energy of the colliding objects. For example, collisions between balls in snooker or pool would typically be highly elastic. A highly inelastic collision dissipates a lot of the energy of the colliding objects. For example, collisions between two blobs of wet clay would typically be highly inelastic.

Friction describes the degree to which the contact resists motion in the plane of the contact surfaces. Friction is always positive. When friction between materials is low, they can slide readily against each other. When friction between materials is high, they tend to grip each other in the plane of their surface contact.

Softness describes the ease with which collision objects will interpenetrate. Softness is always positive. When softness is zero, collision objects do not interpenetrate. When softness is non-zero, objects partially interpenetrate.

Maximum adhesive force describes a force that takes effect when two collision objects touch and acts in a direction that tends to keep the two collision objects together. When the adhesive force is zero, objects bounce off each other readily. When the adhesive force is large, objects tend to cling to one another.

Material interaction parameters are interdependent to some degree. For example, altering the softness parameter has a visible effect on restitution.

Material pairs

The interaction of each pair of materials is defined by one of the material interactions. The materials pair table is part of the virtual world definition which can be edited.

For example, the **Ice-Velcro** material interaction is **Icy**. This means that when a collision object with the Ice material collides with a collision object with the Velcro material, the resulting interaction is defined by the Icy material interaction.

Similarly, when two collision objects of Ice material collide, the resulting **Ice-Ice** interaction is also defined by the **Icy** material interaction.

However, when two collision objects of Velcro material collide, the resulting **Velcro-Velcro** interaction is defined by the **Sticky** material interaction.

Note If you want to change how a particular collision object behaves in general, you would change its **material type**. If you want to change how objects of one particular material type interact with objects of another material type, you could either change the corresponding **material interaction** to set it to another material interaction, or you could edit the properties of that material interaction itself.

Simulation settings

The **simulation settings** are a set of properties that apply to the virtual world and affect how it simulates. The following descriptions summarise the available simulation settings. Many of these settings have additional, detailed help topics in this user guide.

See the **Timeline settings** topic for more information on other settings that affect simulations.

Non-spherical mass objects

You can specify whether the simulation should take into account the actual shape and size of mass objects. This setting does not affect the way mass objects move in a straight line, but it does affect how mass objects rotate.

When this setting is turned off, mass objects are assumed to be spherical, regardless of their actual geometry type. This makes simulations faster but less accurate, because it assumes that mass objects are equally able to rotate about any axis.

When this setting is on, simulations are slower but more accurate. In particular, slender mass objects will be simulated more accurately, as they will rotate more readily about their primary axis than about any other axis.

By default, this setting is turned off, in order to optimise for speed.

Gravity

You can change the direction and gravity of the global gravitational field. The default direction and magnitude correspond to Earth average surface gravity. See the **Gravity** topic for more information.

Material interactions

You can change how collision objects of particular materials interact by modifying the restitution, friction, softness and maximum adhesive force properties of the material interaction that governs their collision. See the **Materials** topic for more information.

Material pairs

You can change how collision objects of particular materials interact by modifying which material interaction model is employed when objects of different material types collide. See the **Materials** topic for more information.

Unit length

You can help *endorphin* to simulate more accurately by specifying a **unit length**, or characteristic length, in metres. This length represents an approximate average size of entities in the virtual world. You can change the unit length to improve the numerical stability of the simulation. This is often called **tuning** the simulation. If your scene was composed of very large characters or scene elements, you might want to increase the unit length to 5.0 or 10.0, for example.

The default unit length is 1.0. This implies that the average size of entities in the scene is around one metre.

Unit mass

You can help *endorphin* to simulate more accurately by specifying a **unit mass**, or characteristic mass, in kilograms. This mass represents an approximate average mass of entities in the virtual world. You can change the unit mass to improve the numerical stability of the simulation. This is often called **tuning** the simulation. If your scene was composed of very large characters or scene elements, you might want to increase the unit mass to 100.0 or 500.0, for example.

Keep in mind that as the unit length increases, the unit mass should increase roughly cubically, since mass is proportional to the cube of the size. For example, if the unit length is 10.0, the unit mass should be around 1000.0.

The default unit mass is 1.0. This implies that the average mass of entities in the scene is around one kilogram.

Timeline settings

The **timeline settings** are a set of properties that apply to the timeline and affect how the virtual world simulates. The following descriptions summarise the available timeline settings.

See the **Simulation settings** topic for more information on other settings that affect simulations.

Frame rate

When you simulate a virtual world, you can choose how often to store snapshots of the scene into the frame buffer by adjusting the **frame rate** setting.

The default frame rate is set to 60 frames per second. With a default simulation frame rate of 120 simulation steps per second, this means that every other frame generated by the physics engine is stored in the frame buffer.

Simulation frame rate

When you simulate a virtual world, a representation of the world is created in a **physics engine**. The engine simulates the physical dynamics of the virtual world by replacing continuous acceleration with discrete timeslices during which velocities are constant and accelerations are zero. The **simulation frame rate** specifies how many discrete physics engine steps to run per second of virtual world time. The higher the number of steps, the more accurate the simulation. Increasing the simulation frame rate may slow the actual speed of the simulation since it becomes more computationally intensive.

The default setting is 120 simulation steps per second. This means that each timeslice during the virtual world simulation is 1/120 of a second in virtual world time. This setting does not affect the simulation rate in terms of real time, which is governed by the complexity of your scene and your available computing power. Increasing the simulation frame rate may slow the actual speed of the simulation since it becomes more computationally intensive.

Simulation steps to run

When you simulate a virtual world, you can stop the simulation at any time by clicking the **Stop** button, or by selecting **Simulation > Play/Stop**. If you do not manually stop a simulation, it will stop automatically when the number of simulation steps specified by the **Simulation steps to run** setting has been reached. This is to avoid inadvertently running a simulation that does not stop.

The default setting is 10000 simulation steps, which is equivalent to 83.3 seconds of virtual world time when simulating at 120 simulation steps per second.

Time magnification

When you replay animation from the frame buffer, you do not need to display it in real time. You can view it faster than real-time, or, more commonly, slow it down to review the motion in slow-motion. A setting of 1.0 implies playback at real-time, whereas a setting of 2.0 implies a 50% slow-motion replay.

You can also set the replay speed by changing the **Replay Speed** setting in the Timeline Editor. Keep in mind that the Replay Speed in the Timeline Editor is the inverse of the Time Magnification setting in the Timeline settings. For example, a time magnification of 2.0 will be displayed as a replay speed of 0.5.

The default time magnification is 1.0.

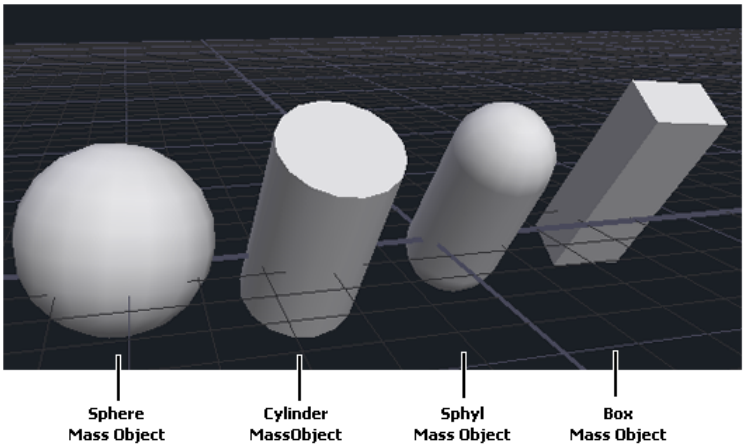
Looping

When you replay animation from the frame buffer, the **Looping** setting specifies whether to continually repeat the replay.

By default, looping is turned on. You can adjust the section of animation played while looping using the **loop range** marker below the frame buffer.

Introducing mass objects

Mass objects represent physical mass. They have a **shape**, **size** and **density**. The combination of volume and density gives the overall **mass** of the object.



Shape and size

Mass objects may be **boxes, spheres**, **cylinders** or **sphyls**.

- **Box** mass objects are box-shaped. You can specify the width, height and length of box objects. Box objects are sometimes called **cuboids** in other applications.
- **Sphere** mass objects are ball-shaped. You can specify the radius of spherical objects.
- **Cylinder** mass objects are tube-shaped with flat ends. You can specify the length and radius of cylinder objects.
- **Sphyl** mass objects are tube-shaped with round ends. You can specify the length and radius of sphyl objects. Sphyl objects are sometimes called **capsules** in other applications. Sphyl is an abbreviation of *sphere and cylinder*.

Density

You can adjust the density of a mass object to change its overall mass. Density is a dimensionless scale factor, where a value of 1.0 implies the density of water (1000.0 kilograms per cubic metre). Most mass objects in the standard *endorphin* simulation character typically have densities in the range 1.5 to 2.0. The default density for new mass objects is 0.75.

It is generally good practice to modify the overall mass of mass objects by changing their size and shape, rather than their density. For example, if you have a large collision object,

and you want to it have a correspondingly large mass, it is usually better to increase the size of its parent mass object to match the size of the collision object, rather than to increase the density of the parent mass object.

The reason for this that if a mass object is small and dense, it will have a tendency to rotate faster than would be realistic. Physically, this is because angular momentum depends on the **distribution** of mass, unlike linear momentum, which depends only on the mass itself. Objects with their mass concentrated near to the axis of rotation will rotate more readily than objects with the same mass distributed far from the axis of rotation.

Forces

Mass objects are influenced by **forces**.

Gravitational force is a global force field. If you add gravity to the virtual world, then any mass objects in the world will experience a downwards force. In *endorphin*, you can you set the magnitude and direction of gravity.

Collision forces occur when objects collide with other objects. Mass objects themselves do not collide directly. Instead, mass objects contain a set of one or more **collision objects**. The collision objects collide, and transmit the corresponding collision forces back to the mass objects.

User-defined forces are optional timeline events that you can add to a scene. For example, you might add a timeline force event to mimic the effect of a gunshot.

Linear momentum

Mass objects have a **velocity**. The combination of the mass and velocity gives the **momentum** of the object. Once a mass object has momentum, it will tend to move in a straight line unless acted upon by some kind of force. The greater the momentum of a mass object, the larger the force that is needed to deviate it from its straight line path.

Regardless of the shape and size of a mass object, the inertial properties of a mass object can be treated as though all its mass is concentrated at a single point at its centre of mass. For example, if you change the shape of a mass object, but keep its mass and centre of mass the same, the resulting inertial properties will be unchanged.

Angular momentum

Mass objects have an **angular velocity**, which measures how fast they are rotating. The combination of the **mass distribution** and angular velocity gives the **angular momentum** of the object. Once a mass object has angular momentum about an axis, it will tend to keep rotating about that axis unless acted on by some kind of **torque**. The greater the angular momentum of a mass object, the larger the torque that is needed to change its rotational velocity or axis of rotation.

Unlike linear momentum, angular momentum not only depends on the mass of the mass object, but also its mass distribution. For example, suppose your virtual world contains two cylindrical mass objects with the same overall mass but different sizes. If a force is

applied to each cylinder through its centre of mass, the cylinders will react in the same manner, moving in the same direction with same velocity and momentum. However, if the forces do not pass through the centres of mass, each cylinder will rotate differently. The long, thin cylinder will readily rotate about its long axis, but will rotate much less readily about any other axis.

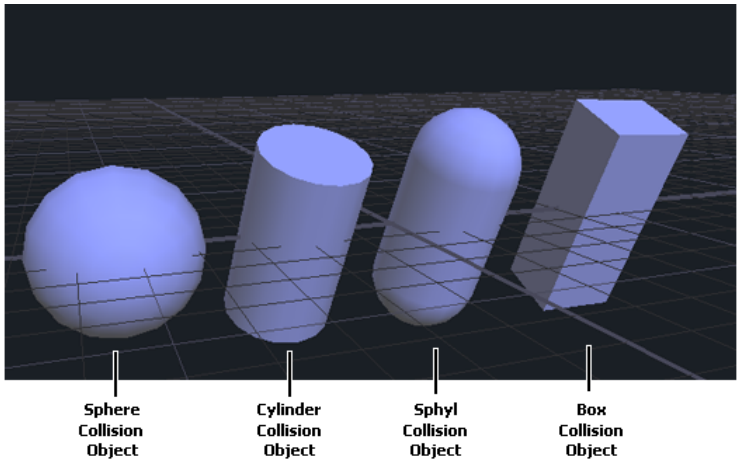
Markers

Each mass object has associated **markers**. Markers are small visualisation objects that are displayed in viewports using the red marker system colour. By default, markers are not displayed. To display markers, select **View > Markers**. Alternatively, **right-click** and ensure Markers is turned on.

Markers were introduced in *endorphin* in conjunction with the Character Studio CSM animation export command. However, this command is no longer extensively used. You will rarely need to display markers.

Introducing collision objects

Collision objects represent surfaces that can be involved in collisions. They have a **shape**, **size** and **material**. Each mass object contains one or more collision objects that define its collision surface shape. Each of the collision objects that are owned by a mass object can have their own shape, size and material.



Shape and size

Collision objects may be **boxes, spheres**, **cylinders** or **sphyls**.

- **Box** collision objects are box-shaped. You can specify the width, height and length of box objects. Box objects are sometimes called **cuboids** in other applications.
- **Sphere** collision objects are ball-shaped. You can specify the radius of spherical objects.
- **Cylinder** collision objects are tube-shaped with flat ends. You can specify the length and radius of cylinder objects.
- **Sphyl** collision objects are tube-shaped with round ends. You can specify the length and radius of sphyl objects. Sphyl objects are sometimes called **capsules** in other applications. Sphyl is an abbreviation of *sphere and cylinder*.

Material

Collision objects have a **material**. Materials affect collisions in two main ways. The material defines the **volume** characteristics of the collision object, such as its compressibility and springiness. The material also defines the **surface** characteristics of the collision object, such as its **friction**.

The default material for new collision objects is **default**.

Most of the collision objects belonging to the standard simulation character have the **skin** material type. This ensures that body parts can move past each other relatively smoothly. The fingers of the standard simulation character have the **sponge** material type, which ensures that the fingers are both soft yet grippy. The heel, ball and toes of the character's feet have the **rubber** material. This ensures that the feet have a grippy, bouncy interaction with the ground.

Collision modes

Collision objects have **collision modes**. Collision modes are used when collision objects are arranged as part of an object hierarchy. You can define whether or not a collision object will interact with its sibling, parent or ancestor collision objects. These modes are very useful when creating characters with many closely-packed collision objects.

Joints

A character is made up of a hierarchy of **joints**. A joint is composed of a **connector** and one or more **bones**. The joint connector is the mechanism that allows the joint to be connected to another joint. Bones are rigid entities that extend from the connector. Each joint can have one or more joints connected to it, forming a hierarchy of joints. Each child joint is connected to one of the bones.

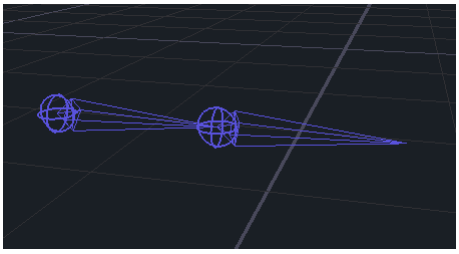
Each joint has one corresponding **mass object**. Each mass object has one or more **collision objects** that define the collision surface for the mass object.

The topmost joint that connects a joint hierarchy to the virtual world is called the **skeleton root joint**.

Skeletal joints

The most common type of joint is the **skeletal joint**. In the viewport, skeletal joints are drawn as three intersecting circles. Skeletal joints allow rotation along all three rotational degrees of freedom. Rotation along the axial direction is commonly referred to as the **twist angle**. Rotations along the two axes perpendicular to the axial direction are commonly referred to as the **swing angles**.

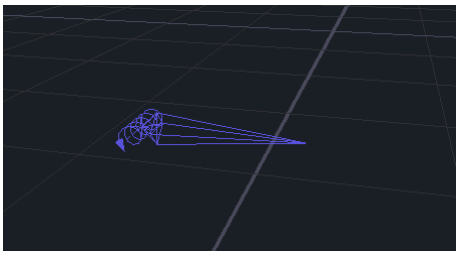
Most joints in humanoid characters are skeletal joints. For example, shoulders, wrists and ankles are modeled using skeletal joints.



Hinge joints

Another common type of joint is the **hinge joint**. In the viewport, hinge joints are drawn as a helix. Hinge joints restrict rotation to a single axis, which is often perpendicular to the bone axis.

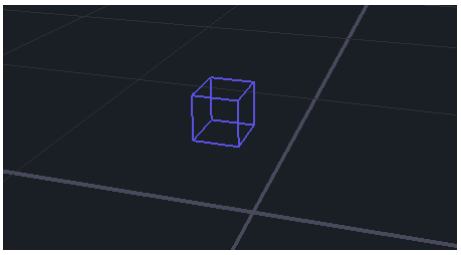
A few joints in humanoid characters are modeled using hinge joints. For example, fingers and knees can be modeled using hinge joints. However, in the *endorphin* standard character, knees are actually modeled using skeletal joints with very small swing limits and a twist limit perpendicular to the axial direction.



Simple joints

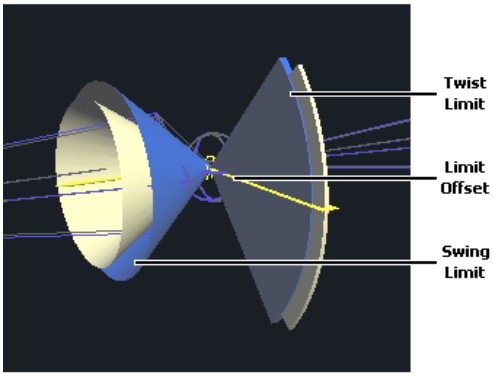
At the end of each joint hierarchy is a **simple joint**. In the viewport, simple joints are drawn as a box. Simple joints do not have a corresponding mass object, and do not contain a connector or a bone.

In a humanoid character, simple joints are placed at the termination of each chain of joints. In practice, this means they are used at the ends of the arms, the ends of the legs, and for the head.



Joint limits

Most joints have a corresponding **joint limit**. The joint limit specifies the range of movement available. Skeletal joints have separate limits for the twist and each of the swing directions. Hinge joints have a limit for the hinge axis. Simple joints are not articulated and so do not have a joint limit.



Hard and soft limits

Joint limits have a **soft limit** and a **hard limit** to specify the allowed range of motion in a more physically accurate way.

When the joint angle is inside the soft limit, the joint can move freely, subject only to the damping force. When the joint is between the soft and hard limits, there is also an additional **stiffness force**, which acts to move the joint back inside the soft limit range. The size of the stiffness force is proportional to how far the joint is between the soft and hard limits. The stiffness force is effectively infinite when the joint is at its hard limit. This prevents a joint from exceeding its hard limit.

Skeletal joints have separate soft limits, hard limits and stiffness properties for the twist direction and for each of the swing directions.

In the viewport, soft limits are displayed in **white**, and hard limits are displayed in **blue**.

Damping

Joint **damping** is a way of specifying how easily the joint is able to move. Damping generates a **damping force** that acts against the direction of joint motion. The size of the damping force is proportional to the speed at which the joint is moving. When damping is low, a joint can move readily. When damping is high, the joint is stiffer, and does not move as readily.

Skeletal joints have separate damping properties for the twist direction and for each of the swing directions.

To display joint limits:

By default, joint limits are not displayed when you create a new scene. This is to avoid cluttering the viewport with graphics that can slow down redraw times and make the viewport difficult to use.

1. Select a joint of interest in the viewport or in the Node View.
2. In the Node View, each joint has a corresponding child node of the same name, except that the **-Joint** suffix is replaced by **-JointLimit**. Select the icon of this child node in the Node View to toggle the visibility of the corresponding joint limit.
3. In the Node View, each joint has a corresponding child node of the same name, except that the **-Joint** suffix is replaced by **-JointLimitOffset**. Select the icon of this child node in the Node View to toggle the visibility of the corresponding joint limit offset.

Note If you are not able to visualise joint limits or joint limit offsets, right-click in the viewport and check that **Joint Limits** is turned on. Alternatively, select **View > Joint Limits** and check that it is turned on.

Problems selecting joint limits

You should be able to select **any** object that appears in the viewport—with the exception of frozen graphical objects and inactive characters in Character Edit Mode. However, some users may find problems selecting **joint limits** using the viewport. If you are having problems selecting joint limits in the viewport, you will need to use the **Node View** to select joint limits instead. This issue may be addressed in future releases of *endorphin*.

Joint limit offsets

Joint limits do not actually directly constrain the range of motion of their corresponding child joints. Rather, they constrain the range of their corresponding **joint limit offset**. The child joint itself has a relative offset to the joint limit offset. Hinge joints have a single hinge joint limit offset. Swing joints have both swing and twist limit offsets.

Joint limit offsets are displayed in **yellow**. Swing offsets are drawn using a needle-shaped graphic. Twist offsets is displayed as a flat bar graphic.

To display joint limit offsets:

By default, joint limit offsets are not displayed when you create a new scene. This is to avoid cluttering the viewport with graphics that can slow down redraw times and make the viewport difficult to use.

1. Select a joint of interest in the viewport or in the Node View.
2. In the Node View, each joint has a corresponding child node of the same name, except that the **-Joint** suffix is replaced by **-JointLimit**. Select the icon of this child node in the Node View to toggle the visibility of the corresponding joint limit.
3. In the Node View, each joint has a corresponding child node of the same name, except that the **-Joint** suffix is replaced by **-JointLimitOffset**. Select the icon of this child node in the Node View to toggle the visibility of the corresponding joint limit offset.

Note If you are not able to visualise joint limits or joint limit offsets, right-click in the viewport and check that **Joint Limits** is turned on. Alternatively, select **View > Joint Limits** and check that it is turned on.

Graphical objects

Each mass object can have one or more associated **graphical objects**. A graphical object does not participate in simulations in any way. Instead, graphical objects are used to help visualise characters.

The standard *endorphin* character has a **HeadGraphic** graphical object associated with the **HeadMass** mass object. The HeadGraphic object helps make the orientation of the *endorphin* character more obvious.

Simulation characters also have small graphical objects that indicate the position of the left and right **elbows** and **knees**. These objects are useful when examining arm and leg rotations. These are the **LeftElbowGraphic**, **RightElbowGraphic**, **LeftKneeGraphic** and **RightKneeGraphic** objects respectively.

In the current version of *endorphin*, these helper graphical objects are **always** added to characters in scenes, even if you explicitly delete these objects from the corresponding character definitions in Character Edit Mode. This behaviour may change in future releases of *endorphin*.

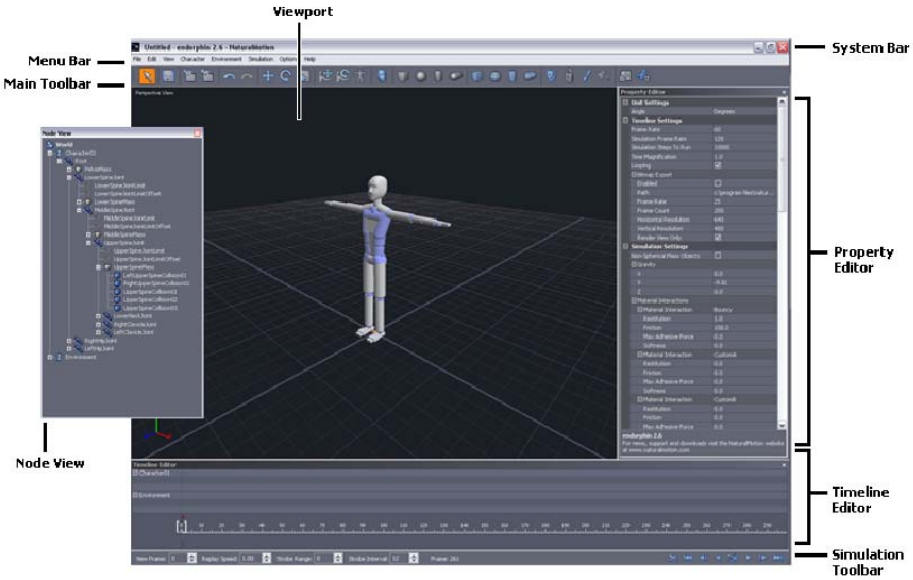
If you do not want to see these graphical objects, you can **hide** them from the scene using the **Node View**. Right-click on the graphic objects that you want to hide, and then right-click and select **Hide** from the context menu. These visibility settings are stored in the scene file and are used when the scene file is reopened later.

Chapter 3

User Interface

Introducing the user interface

endorphin is a 3D application designed to generate animation as part of an animation pipeline. You may be familiar with a lot of the *endorphin* user interface and workflow, as the system has been designed to work in a similar manner to existing animation systems:



Application window

The *endorphin* application window contains menus, toolbars and tool windows.

To launch *endorphin*:

There are a number of ways to launch *endorphin*:

- In the **Windows File Explorer**, locate the file **endorphinApp.exe**. This file is located in the **bin** folder. Double-click on this file icon. Alternatively, double-click on the file **endorphin.exe** to launch *endorphin* without the initial splahscreen.
- Alternatively, use to the Windows **Start** menu launch *endorphin*. **Select Start > All Programs > NaturalMotion**, and browse to select the ***endorphin*** submenu.
- Alternatively, if you have created an *endorphin* **desktop icon**, double-click on this desktop icon to launch *endorphin*. By default, a desktop icon is created when you install *endorphin*.
- Alternatively, if you have created an *endorphin* **QuickLaunch icon**, double-click on this QuickLaunch icon in the Windows TaskBar to launch *endorphin*. By default, a QuickLaunch icon is created when you install *endorphin*.

To launch *endorphin* using a scene file:

You can also launch *endorphin* by double-clicking on an *endorphin* **.ens** scene file icon in Windows File Explorer or on the desktop. A **new instance** of endorphin is launched, with the selected scene file preloaded. Keep in mind that if you have multiple versions of endorphin installed, the **most recently installed** version of endorphin will be associated with the **.ens** file extension.

To resize *endorphin*:

You can use the *endorphin* standard Windows title bar to minimise, maximise and restore the application. These commands are also available by right-clicking on the title bar. Also, you can double-click on the title bar to restore and maximise the application.

To close *endorphin*:

To close *endorphin*, click on the **Close** button [X] in the title bar. Alternatively, select **File > Exit**. The keyboard shortcut is **Ctrl+F4**.

Full Screen mode

To display *endorphin* in **Full Screen** mode, select **View > Full Screen**. The keyboard shortcut is **Alt+F**. In Full Screen mode, the application frame, application title bar and Main Toolbar are all hidden. Once endorphin is in Full Screen mode, the only way to turn Full Screen mode off is to use the keyboard shortcut, since the Main Toolbar is hidden.

In order to use tool windows in Full Screen mode, they must be floating rather than docked. Docked tool windows are hidden in Full Screen mode.

Multiple monitors

endorphin supports the use of multiple monitors. You can stretch the main application window over two or more monitors. Alternatively, you might wish to work with the application window on one monitor and the tool windows floating on a second monitor.

Working folders

When you are using *endorphin*, you will often need to browse for files and folders. To minimise the amount of repetitive file browsing, *endorphin* keeps track of the working folder for a range of operations. These working folders are stored in the **Windows Registry** as an application-wide setting. Each time *endorphin* is launched, it initalises its working folders based on the working folders stored by the previous session of *endorphin*.

Working folders stored by *endorphin*

endorphin stores separate working folders for:

- **Scenes**: Used when opening and saving scene files.
- **Characters**: Used when creating, opening and saving character files in Character Edit Mode; also used when adding characters to scenes.
- **Import**: Used when importing animation data, animated cameras and OBJ mesh files.
- **Export**: Used when exporting animation data and AVI movie files.
- **Environments**: Used when saving and loading the Environment character.
- **Poses**: Used when saving and loading poses and active poses.
- **Viewports**: Used when saving and loading viewport files.

Character working folder

The **Character** working folder is displayed at the bottom of the **New Character**, **Open Character**, **Save Character and Add Character** dialogs, along with a **Browse** button so that you can change the character working folder.

These dialogs display the characters found in the specified character working folder. *endorphin* lists the simulation and prop characters found in the folder in two separate lists. In addition, the **New Character**, **Open Character** and **Add Character** dialogs also include the Standard Simulation Character in addition to the characters located in the character working folder.

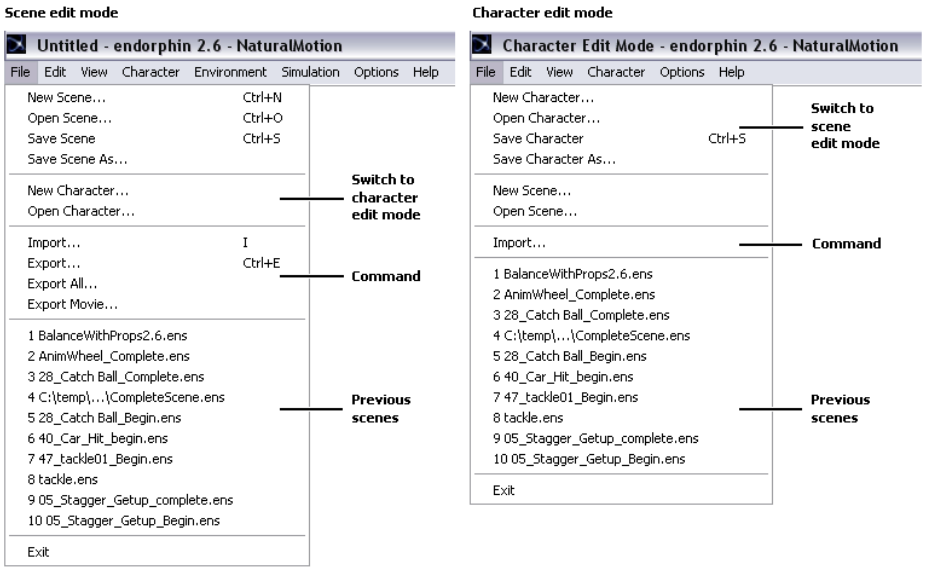
All the character dialogs include a **Browse** button so that you can change the character working folder.

Menu Bar

The menu bar contains most of the commands you will need to use with *endorphin*. Where a command has a corresponding keyboard shortcut, the shortcut is displayed alongside the command name. Commands that are followed by **…** indicate that they display a dialog or editor. The menu bar is always visible.

Most commands also have a corresponding **accelerator shortcut**. Press the **Alt** key to display accelerator shortcuts in the Menu Bar. Press **Alt** again to hide the accelerator shortcuts. Accelerator shortcuts are displayed as **underlined** characters in command names. When accelerator shortcuts are displayed, you can navigate the Menu Bar by pressing the accelerator shortcut keys directly. For example, to open a scene using accelerator shortcuts, press **Alt**, then press **F** to display the File command menu, and then press **O** to run the **Open Scene…** command.

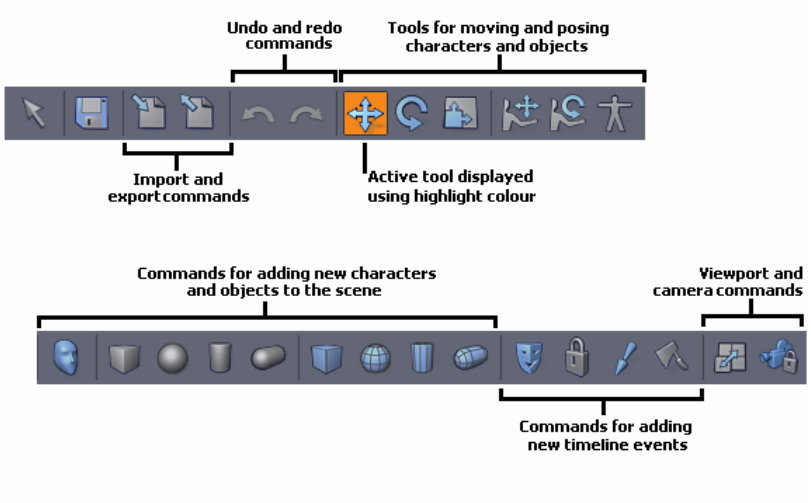
You can also navigate the Menu Bar using the arrow keys.



Main Toolbar

The main toolbar contains buttons for many of most commonly used *endorphin* tools and commands. You can toggle the visibility of the main toolbar by selecting **View > Main Toolbar**. If you hover over a button in the Main Toolbar, a tooltip will display the corresponding command name.

The Main Toolbar is designed to show all the toolbar icons at resolutions of 1024 x 768 pixels and higher. If you are running *endorphin* on a lower-resolution monitor, you might find that not all the buttons on the Main Toolbar are visible. Also, if you dock the Main Toolbar vertically rather than horizontally, you might find that not all its buttons are visible.

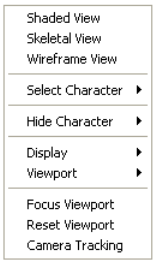


Context menus

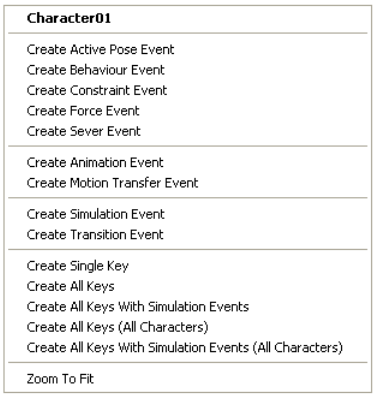
In common with other Windows applications, *endorphin* often displays a **context menu** in response to a **right-click**.

Many of the tool windows, such as the Node View, Timeline Editor and Property Editor, contain commands in their context menus. Similarly, most viewport functionality is available in the viewport context menu.

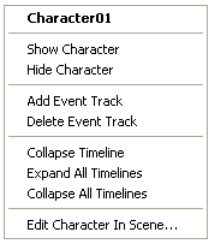
To close a context menu without selecting an item, **click** anywhere outside the context menu. Alternatively, press **Esc** to close the context menu.



Viewport Context Menu
To display, right-click in a viewport



Timeline Editor Context Menu
To display, right-click on character timeline in Timeline Editor



Timeline Editor Character Context Menu
To display, right-click on character name in Timeline Editor

Getting help

This ***endorphin* User Guide** is available by selecting **Help > endorphin Help**. The keyboard shortcut is **F1**. You can view the User Guide without having to run *endorphin*. To launch the User Guide without starting *endorphin*, browse to the **bin** folder in the installation folder, and double-click on the **endorphin.chm** Compiled HTML Help file.

The ***endorphin* User Community Forum** is a useful site for news, information, tutorials, with an active user community. To navigate to the *endorphin* User Forum, select **Help > endorphin Forum**. This opens your default browser and navigates to the User Forum homepage.

The **Getting Started** dialog appears when you first launch *endorphin*. It contains a number of useful links to various resources, such as tutorials, the User Forum and the NaturalMotion homepage. To display the Getting Started dialog, select **Help > endorphin Welcome**.

The **About Box** dialog contains some useful information such as the build number of your *endorphin* system. If you ever have problems with *endorphin*, your build number may be useful for technical support. To display the About Box dialog, select **Help > About endorphin…**.

About Box

You can display the *endorphin* **About Box** by selecting **Help > About endorphin…**. The About Box is contains information about your installation of *endorphin*, including the build number, hardware lock expiration date, and a link to the NaturalMotion homepage.

Build number

All *endorphin* builds have an **internal build number** in addition to the published version number. For example, the internal build number of *endorphin* 2.6.0 is **2.6.0.6126**. When reporting technical problems or issues, it can be useful to quote the internal build number when communicating with NaturalMotion or the *endorphin* User Community Forum.

The build number can be useful for distinguishing many installations of *endorphin*—such as different editions, major releases or point releases—that may be installed on your system. In particular, the build number is useful to know if you have been using a **pre-release version** of *endorphin*, such as an Alpha or Beta version. Issues are often resolved between the pre-released and released versions of *endorphin*.

Hardware lock expiration date

If you are using a licensed version of *endorphin*, *endorphin* Educational Edition or *endorphin* Learning Edition, you can use *endorphin* for an unlimited period of time.

However, if you are using an **evaluation license** of *endorphin* or *endorphin* Educational Edition—or if you are using a rental licensing scheme such as *endorphin* **Rush**—then your license of *endorphin* will usually have an expiration date after which it will no longer operate.

You can find the expiration date of your license by selecting **Help > About endorphin…** to display the **About NaturalMotion *endorphin*** dialog. This dialog displays your hardware lock expiration date. If this textbox is empty, you are using a time-unlimited license.

Introducing tool windows

Property Editor

The Property Editor displays properties for the selected entity or entities. If no entities are selected, the Property Editor displays properties for the virtual world as a whole. You can toggle the visibility of the Property Editor by selecting **View > Property Editor**. The Property Editor is displayed by default.

Timeline Editor

The Timeline Editor displays the time-based events that are triggered as a scene simulates. Each character in the scene has a separate character timeline in the Timeline Editor, made up of multiple timeline tracks. You can toggle the visibility of the Timeline Editor by selecting **View > Timeline Editor**. The Timeline Editor is displayed by default.

Main Toolbar

The Main Toolbar displays a set of command **buttons** for commonly-used commands. You can toggle the visibility of the Main Toolbar by selecting **View > Main Toolbar**. The Main Toolbar is displayed by default.

Node View

The Node View displays a hierarchical representation of the entities in a scene. The Node View is a useful way to select entities in the scene, particularly where they are difficult to select graphically. You can toggle the visibility of the Node View by selecting **View > Node View**. The keyboard shortcut is **N**. The Node View is not displayed by default.

Motion Transfer Editor

The Motion Transfer Editor displays motion transfer settings for the selected character, such as rig node connections and strengths. These settings are used when you are importing and exporting motion data as part of an animation pipeline. You can toggle the visibility of the Motion Transfer Editor by selecting **View > Motion Transfer Editor**. The keyboard shortcut is **M**. The Motion Transfer Editor is not displayed by default, except when entering Character Edit Mode.

Simulation Toolbar

The Timeline Editor also contains the Simulation Toolbar, which is made up of a number of buttons that control common time-based commands such as Simulate, Play and Stop. If you hover over a button in the Simulation Toolbar, a tooltip will display the corresponding command name. Note that the Simulation Toolbar cannot be separated from the Timeline Editor.

Scene Notes

The Scene Notes is an editor that displays formatted text that is stored on a per-scene basis. You can use the Scene Notes to store information about the history of a scene, or to share information among team members. You can toggle the visibility of the Scene Notes by selecting **View > Scene Notes**. The Scene Notes are not displayed by default.

Output

The Output displays text-based output of certain commands. You will rarely need to use the Output, and it is hidden by default. You can toggle the visibility of the Output by selecting **View > Output**.

Working with tool windows

The main **tool windows** are the Main Toolbar, Property Editor, Node View, Timeline Editor, Motion Transfer Editor, Scene Notes and Output. These windows can be either floating or docked. By default, the Node View and Scene Notes are floating, and the Property Editor, Timeline Editor, Motion Transfer Editor, Main Toolbar and Output are docked. However, you can change the arrangement of the tool windows to suit your working style. Each time *endorphin* is launched, it restores the arrangement of tool windows from the previous session.

Floating the tool windows is useful, especially if you have more than one monitor. For example, when working with complicated scenes, it is useful to float tool windows—in particular, the Node View, Timeline Editor, Motion Transfer Editor and Scene Notes—and move them to a second monitor. This allows you to increase their size without affecting the viewports.

Docking the tool windows is useful when you want to keep the tool window layout neat and tidy. The Node View, Property Editor, Motion Transfer Editor, Scene Notes and Output can be docked to the left edge or the right edge of the main application window. The Timeline Editor can be docked to the top edge or the bottom edge of the main application window. The Main Toolbar can be docked to any of the top, bottom, left or right edges.

To float a docked tool window:

Click anywhere on the window titlebar and drag it away from its docked location. Alternatively, simply **double-click** the window titlebar.

To dock a floating tool window:

Click anywhere on the tool window frame, and drag it over the main application window. When you are in a potential dock location, *endorphin* will display a rectangle indicating

this position. Alternatively, double-click on the tool window frame to automatically dock the window.

To prevent a floating tool window from docking:

Hold down the **Ctrl** key to disable docking when you are dragging a tool window.

By default, when you drag a floating tool window, it will automatically attempt to dock itself if you drag it close to a potential docking position. However, you may want to prevent this behaviour if you want to move the tool window to a position where it would otherwise attempt to dock itself.

To hide a floating tool window:

Click the **Close** button [X] in the window titlebar to hide the window.

To prevent docked windows from floating:

Once you have a tool window layout that you are satisfied with, you may not want to inadvertently float tool windows that have been docked. To lock the layout, select **View > Lock Layout**. The keyboard shortcut is **Ctrl+Shift+L**. When the layout is locked, you can dock floating tool windows, but you cannot float docked tool windows. Keep in mind that even when the layout has been locked, you can still resize docked windows.

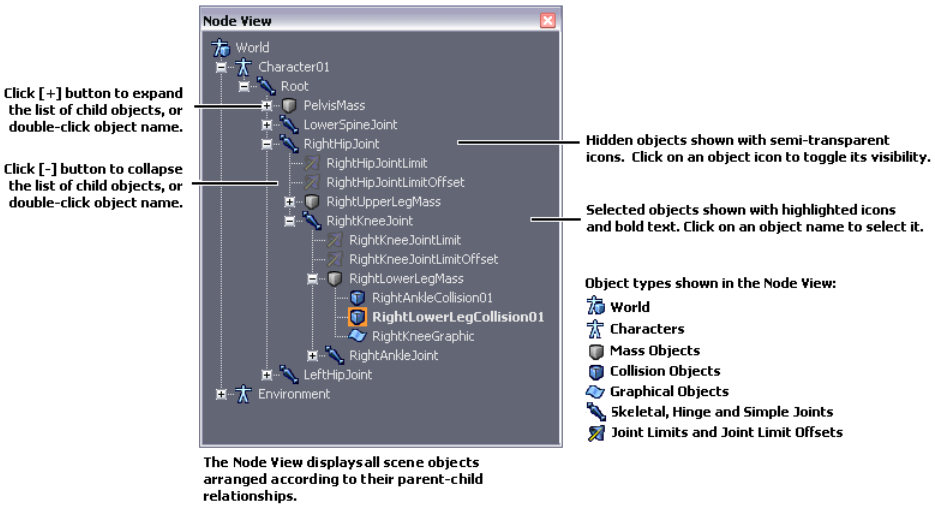
To reset the layout:

If you want to reset the layout of the tool windows back to the shipped *endorphin* defaults, select **View > Reset Layout**. This is useful if you are having trouble getting tool windows to dock, for example, of if you have lost a tool window outside the bounds of your monitors.

Node View

The **Node View** displays a hierarchical representation of the entities in a scene. The Node View is a useful way to select entities in the scene, particularly where they are difficult to select graphically.

You can toggle the visibility of the Node View by selecting **View > Node View**. The keyboard shortcut is **N**.



Using the Node View

To expand or collapse objects in the Node View:

- The Node View displays the objects in the scene in a hierarchical arrangement. This helps you identify of characters and the corresponding parent-child relationships. If an entity in the Node View is **expanded**, this means its children are displayed in the Node View. If an entity is collapsed, this means its children (and any of their children, and so on) are not displayed in the Node View.
- To expand or collapse an item in the Node View, **double-click** on it. Alternatively, click the **[+]** buttons to expand collapsed items, and click the **[-]** buttons to collapse expanded items.
- Prior to *endorphin* 2.6.1, adding or removing characters or other objects from a scene would result in all the nodes in the Node View resetting to their **collapsed** state. Starting with *endorphin* 2.6.1, adding or removing objects causes the

corresponding nodes to be added or removed from the Node View without affecting the expanded or collapsed state of **other** nodes in the Node View.

To select objects using the Node View:

- To select an object using the Node View, click on the **name** of the entity. This will select that entity, and deselect all other entities in the scene. The Node View does not support additive selection. Be careful not to click on the icon associated with the entity, as this will toggle the **visibility** of the item, rather than its selection state.
- The **Select** tool does not need to be active in order to select items using the Node View.
- You can select any item in the Node View, even it is has been hidden.

To show and hide objects using the Node View:

- To show or hide an object using the Node View, click on the **icon** next to the name of the entity. When the entity is visible, its icon is drawn solidly in the Node View. When the entity is hidden, its icon is drawn partially transparently. Each object stores its visibility setting in the scene file. These visibilities are restored when the scene is reopened.

To show and hide objects using the Node View context menu:

If you right-click on an entity in the Node View, the context menu offers more commands for showing and hiding objects. These commands are useful when you want to show or hide objects and also their child objects as well, in a **recursive** manner down the object hierarchy. Each object stores its visibility setting in the scene file. These visibilities are restored when the scene is reopened.

- Select **Show** to show the entity. Keep in mind that if you Show a character, only its character cube is shown.
- Select **Hide** to hide the entity. Keep in mind that if you Hide a character, only its character cube is hidden.
- Select **Show All** to show the entity, along with all of its child entities. The Show All command is a convenient way of showing an entire character.
- Select **Hide All** to hide the entity, along with all of its child entities. The Hide All command is a convenient way of quickly hiding an entire character.
- Select **Show Type > Mass Objects** to show the child mass objects of an entity. There are similar commands to show Collision Objects, Graphical Objects, Joints and Joint Limits.

- Select **Hide Type > Mass Objects** to hide the child mass objects of an entity. There are similar commands to show Collision Objects, Graphical Objects, Joints and Joint Limits.

Note Joint limits are a special case, and are not affected by the Show All or Hide All commands. This is so that you can use the Show All and Hide All commands in conjunction with characters without having joint limits being shown when you show the character.

To freeze and unfreeze graphical objects using the Node View:

It is often useful to temporarily prevent a graphical object from being selectable. This is particularly the case when you are filling out a graphical object that represents environment geometry. It is also useful when you are reshaping a character in Character Edit Mode to fit a skin represented by a graphical object. When a graphical object is **frozen**, it cannot be selected in the viewport. However, it can still be selected using the Node View. Note that each time a scene is reloaded, all graphical objects are unfrozen.

If you right-click on a graphical object in the Node View, the context menu offers more commands for freezing and unfreezing graphical objects..

- Select **Freeze** to freeze an unfrozen graphical object.
- Select **Unfreeze** to unfreeze a frozen graphical object.

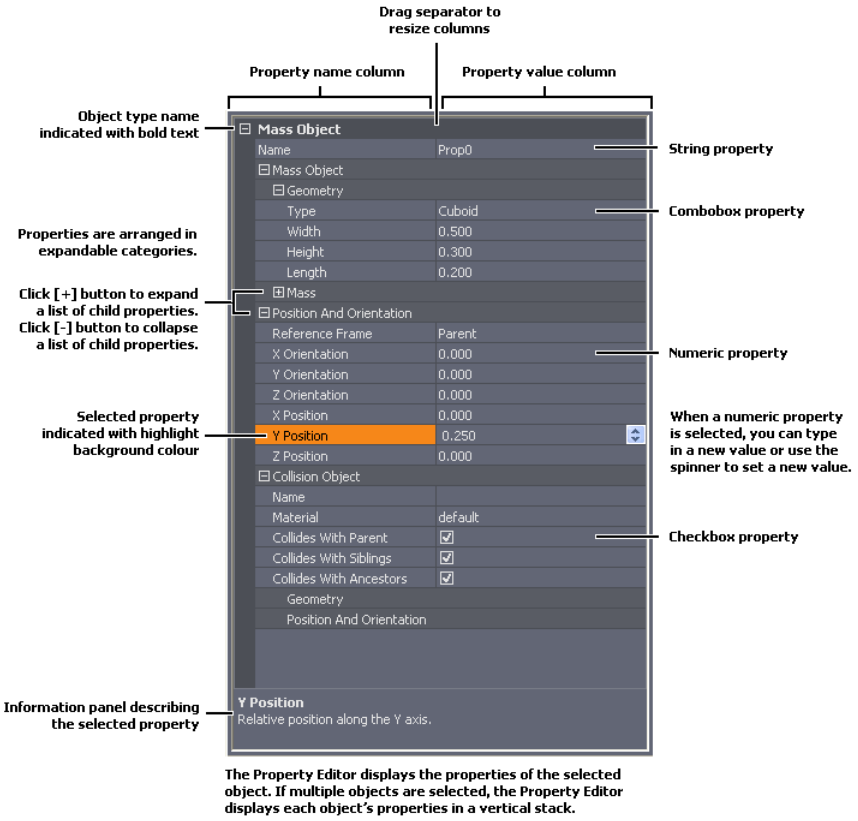
Property Editor

The **Property Editor** is the central tool window for viewing and modifying the properties of scene and timeline entities. The Property Editor displays a **property sheet** for selected entities. The property sheet is made up two columns—the left hand column containing the name of a property, and the right-hand column containing its corresponding value.

If no entities in the scene or timeline are selected, the Property Editor displays a property sheet containing **simulation** settings, such as gravity and friction properties, and **timeline** settings, such as the simulation frame rate and playback rate.

If one or more entities in the scene or timeline are selected, the Property Editor displays a property sheet for each selected entity. All scene entities, such as mass objects, collision objects and joints, as well as all timeline events, such as force and behaviour events, have corresponding property sheets that are displayed in the Property Editor.

You can toggle the visibility of the Property Editor by selecting **View > Property Editor**.



The Property Editor displays the properties of the selected object. If multiple objects are selected, the Property Editor displays each object's properties in a vertical stack.

Using the Property Editor

To select properties in the Property Editor:

- To edit a property in the Property Editor, **select** the row containing the property. When that property is selected, it is highlighted using the orange highlight colour. Most selected properties have a property **description** that is displayed at the bottom of the Property Editor.
- To select the **previous** property, the keyboard shortcut is **UpArrow**.
- To select the **next** property, the keyboard shortcut is **DownArrow**.
- To select the **first** property, the keyboard shortcuts are **Home** or **PgUp**.
- To select the **last** property, the keyboard shortcuts are **End** or **PgDown**.

To edit properties in the Property Editor:

- To edit a property in the Property Editor, first **select** the row containing the property.
- Some properties are **numeric** properties. To modify a numeric property, **type** in a new value, such as "-12.3" or "3". If the numeric property is an integer, such as the Simulation Frame Rate, then the value you enter will be rounded to the nearest whole number. If the numeric property is limited to a range, such as 0.0 to 1.0, then the value you enter will be limited by this range.

  When numeric properties are selected, a spinner control is displayed alongside the property value. Click the **Up** arrow of the spinner to increase the numeric value using fixed steps. Click the **Down** arrow of the spinner to decrease the numeric value using fixed steps. You can also **click and drag** the Up or Down arrows to rapidly change the numeric value using fixed steps. Each numeric property field has its own increment size, but typically the increment will be around 5% to 10% of the property value.
- Some properties are **combobox** properties. This means that the property value must be one of a fixed range of values. For example, the geometry type of a mass object must be one of Box, Cylinder, Sphere or Sphyl. When a combobox property is selected, a combobox is displayed in the value column. **Click** anywhere in the value column to expand the combobox drop-down list, and click again to make a selection. Click **Esc** to hide the drop-down list without making any changes.
- Some properties are **checkbox** properties. This means that the property value must be either turned on or off. To change the value of a checkbox property, **click** anywhere in the value column.

- Some properties are **path** properties. This means that the property value defines the name of a folder. You cannot type in a new value for path properties. Instead, select the path property to display a **[…]** button. Click this button to display the **Browse For Folder** dialog.
- Some properties are **collection** properties. This means that they contain a **set** of one or more objects. Usually, collections must contain objects of a particular type that depends on the specific property. For example, the Target Objects property of a force event is a collection property made up of a set of mass objects to which the force applies.

  To edit a collection property, click the **[Select]** command hotlink to enter Selection mode. This lets you select a new set of entities. If a **[Clear]** command hotlink is available, click it to clear the current collection.
- Some properties are **read-only** properties. This means that you can view the property, but not edit it. For example, you cannot edit the Name property for any of the objects that make up the standard simulation character.

To expand or collapse groups of properties in the Property Editor:

- The Property Editor often displays sets of related properties as **groups** that can be expanded or collapsed. Also, the entire property sheet of a single selected entity is also a group that can be expanded or collapsed. By default, property groups are expanded.
- To collapse an expanded group, **click** on the **[-]** icon next to the property group name. Alternatively, **double-click** anywhere on the property group name. Alternatively, select the property group and press the **Enter** key.
- To expand a collapsed group, **click** on the **[+]** icon next to the property group name. Alternatively, **double-click** anywhere on the property group name. Alternatively, select the property group and press the **Enter** key.

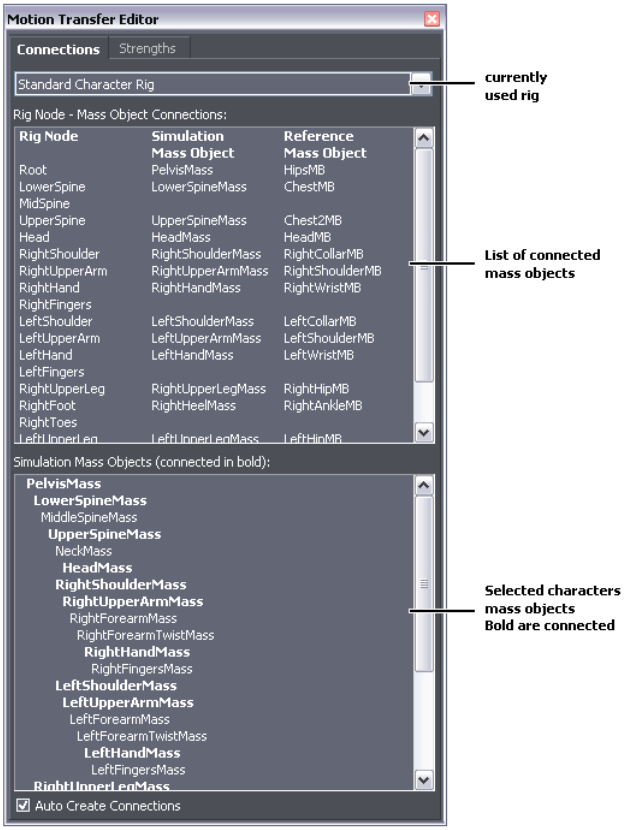
To resize columns in the Property Editor:

- Hover the mouse over the vertical line that separates the property **name** column from the property **value** column. When the cursor changes to the left-right cursor, click and drag the mouse to change the relative sizes of the two columns. If you resize the width of the Property Editor, the name and value columns resize to maintain their relative widths.

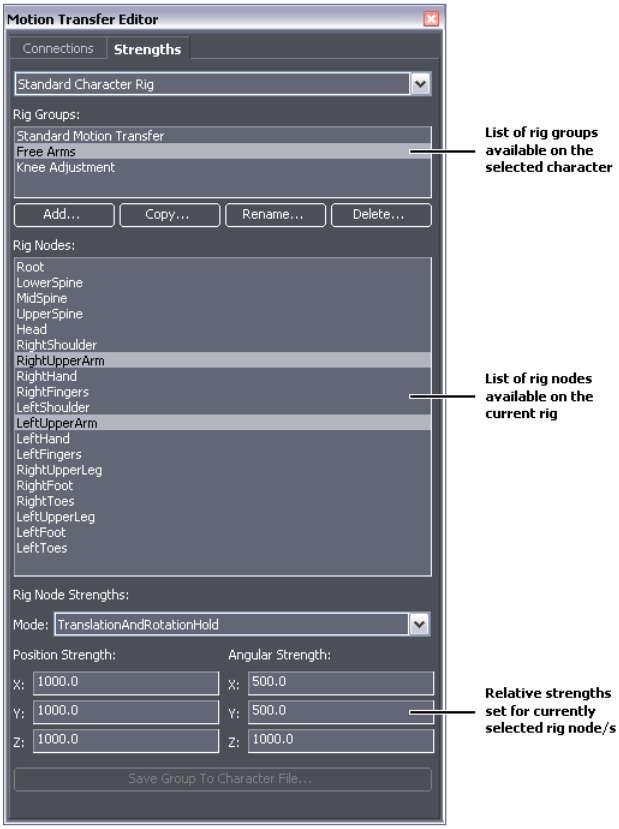
Motion Transfer Editor

The **Motion Transfer Editor** is the central tool window for viewing and modifying the rig node connections and strengths of a character. The Motion Transfer Editor is composed of a **Connections** page and a **Strengths** page.

The **Connections** page allows you to connect a character's mass objects to rig nodes. All characters must be **rigged** to rig nodes in order to be able to import and export animation using dynamic motion transfer. The **Auto Create Connections** setting on the Connections page specifies whether or not to automatically create rig node connections when snapping simulation characters to reference characters in Character Edit Mode.



The **Strengths** page allows you to add, copy, rename and delete **rig groups**. Each rig group is a collection of rig nodes and their corresponding hold mode, positional strength and angular strength. If you are editing rig node strengths of a character in a scene, you can copy these changes to its source character definition file using the **Save Group To Character File…** button.



You can toggle the visibility of the Motion Transfer Editor by selecting **View > Motion Transfer Editor**. The keyboard shortcut is **M**.

Using the Motion Transfer Editor

In Character Edit Mode, you can use the Motion Transfer Editor to edit motion transfer **connections** and **strengths**. The Motion Transfer Editor always displays data for the **active** character. You can edit simulation characters or reference characters. However, you will usually only edit rig node connections for **reference** characters.

When editing scenes, you can use the Motion Transfer Editor to edit character motion transfer **strengths**. You do **not** need to place the character into Edit Character In Place mode to do this. Rather, simply **select** the character that you want to edit. The Motion Transfer Editor uses **sticky** selection. This means that once you have selected a character and displayed its data in the Motion Transfer Editor, you can deselect the character without losing its data from the editor. Note that you cannot edit rig node connections in scenes.

To edit character rig node connections:

1. **Open** the character that you want to edit in Character Edit Mode.
2. **Activate** the character.
3. Browse to the **Connections** tab.
4. Ensure that the **Standard Character Rig** is selected in the Character Rig combobox.
5. Ensure that the **Auto Create Connections** setting is turned **off**. This setting should be turned **on** when you are creating or editing character rig settings using the Move tool.
6. Select a rig node from the **Rig Node Connections** list. If it is connected to a mass object, this mass object is selected in the **Mass Objects** list. Alternatively, select a mass object from the Mass Objects list. If it is connected to a rig node, this rig node is selected in the Rig Node Connections list.
7. If the selected rig node is connected to a mass object, the **Disconnect** button will be enabled. Click the Disconnect button to disconnect the selected rig node from its mass object.
8. If the selected rig node is not connected to a mass object, then select a mass object that is unconnected. The Connect button will be enabled. Click the Connect button to connect the selected rig node to the selected mass object.

To edit character rig node strengths:

1. **Open** the character that you want to edit in Character Edit Mode. Alternatively, open a scene containing the character. Keep in mind that any changes you make to a character in a scene will not affect the corresponding character definition.
2. **Activate** the character (in Character Edit Mode) or select the character.
3. Browse to the **Strengths** tab.
4. Ensure that the **Standard Character Rig** is selected in the Character Rig combobox.
5. Select the **rig groups** that you want to modify. Hold down the **Ctrl** key to select multiple rig groups. Keep in mind that you cannot edit the Standard Motion Transfer character.
6. Select the **rig nodes** that you want to modify. Hold down the **Ctrl** key to select multiple rig node.
7. Change the **Mode** combobox to change the **Hold Mode** of all selected rig nodes in all selected rig groups.
8. Edit the **Position Strength** and **Angular Strength** values to change the rig node strengths of all selected rig nodes in all selected groups.

To add, copy, rename and delete rig groups:

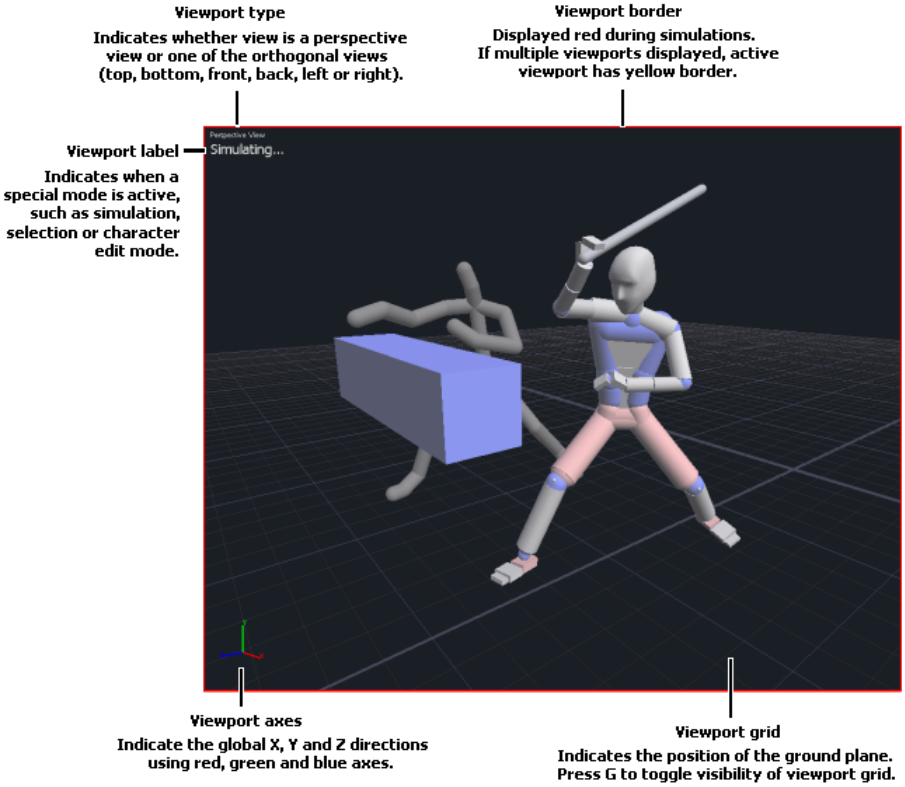
1. **Open** the character that you want to edit in Character Edit Mode. Alternatively, open a **scene** containing the character. Keep in mind that any changes you make to a character in a scene will not affect the corresponding character definition.
2. **Activate** or **select** the character.
3. Browse to the **Strengths** tab.
4. Ensure that the **Standard Character Rig** is selected in the Character Rig combobox.
5. You can make the following types of change:
   - Click the **Add…** button to create a new rig group.
   - Click the **Copy…** button to create a new rig group using a copy of the selected rig group. This button is only enabled when a single rig group is selected.
   - Click the **Rename…** button to rename the selected rig group. You cannot rename the **Standard Motion Transfer** rig group. This button is only enabled when a single rig group is selected. All rig groups must have unique names.
   - Click the **Delete…** button to delete the selected rig groups. You cannot delete the **Standard Motion Transfer** rig group.

To copy motion transfer settings into a character definition file:

1. Open a scene containing the character. Select the character and make the desired changes to its rig groups and rig nodes.
2. **Activate** or **select** the character.
3. Browse to the **Strengths** tab.
4. **Select** one or more rig groups that you would like to copy into its character definition **.nmc** file.
5. Click the **Save Group To Character File…** button to copy the selected rig groups. Keep in mind that this command is not available if you are editing an instance of the Standard Simulation Character in a scene (since this character cannot be modified).
6. The selected rig groups are copied to the character file. If you have created any new rig groups, these will be copied into the character file, along with any changes to existing rig groups.

Viewports

These are 3D views of the *endorphin* virtual world. You will carry out most scene editing tasks in the viewports. You can view a single viewport, or an array of four viewports. At least one viewport is always visible.



Working with viewports

The main *endorphin* window contains the **viewports**. A viewport is a 3D view of the virtual world. You will perform most of your scene editing in *endorphin* by working with viewports.

To rotate a viewport:

- Hold down the **middle mouse button**. Drag the mouse in any direction to rotate the viewport in that direction.

- Alternatively, press **Alt** and the hold down the **left mouse button**. Drag the mouse in any direction to rotate the viewport in that direction.

To zoom a viewport:

- Rotate the **mouse wheel** to zoom the viewport display using incremental steps. Roll up to zoom in. Roll down to zoom out.
- Alternatively, press **Alt** and hold down the **right mouse button**. Drag the mouse to perform a smooth zoom. Move right or down to zoom in. Move left or up to zoom out.

To pan a viewport:

- Hold down both the **middle mouse button** and **right mouse button**. Drag the mouse to pan the viewport in that direction.
- Alternatively, press **Alt** and hold down the **middle mouse button**. Drag the mouse to pan the viewport in that direction.
- Alternatively, press **Ctrl+Alt** and hold down the **left mouse button**. Drag the mouse to pan the viewport in that direction. This is useful if your mouse does not have a middle mouse button.

To focus the viewport:

- Select **View > Focus Viewport** to pan the scene so that the world origin is at the middle of the screen. This is called **focusing** the viewport. Alternatively, **right-click** in the viewport and select **Focus Viewport**. The keyboard shortcut is **Ctrl+F**.
- If you have one or more entities selected, the **Focus Viewport** command pans the scene so the centre of the selected entity set at the centre of the screen. This is a very useful technique. If you have a particular entity of interest that you want to see more clearly, select it and run the Focus Viewport command. You can then zoom in and out, with the entity maintaining its same position in the middle of the screen.

Problems selecting joint limits

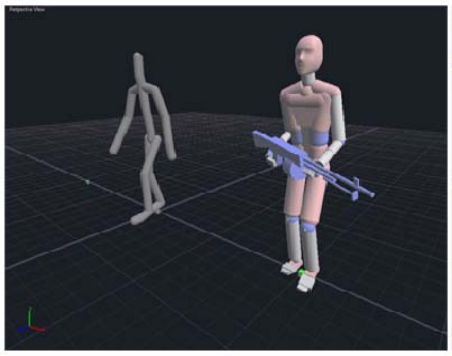
You should be able to select **any** object that appears in the viewport—with the exception of frozen graphical objects and inactive characters in Character Edit Mode. However, some users may find problems selecting **joint limits** using the viewport. If you are having problems selecting joint limits in the viewport, you will need to use the **Node View** to select joint limits instead. This issue may be addressed in future releases of *endorphin*.

Working with multiple viewports

When you first create a scene, *endorphin* displays a single viewport. However, you can also work with multiple viewports. This is often useful when you want to view your scene from different orientations at the same time.

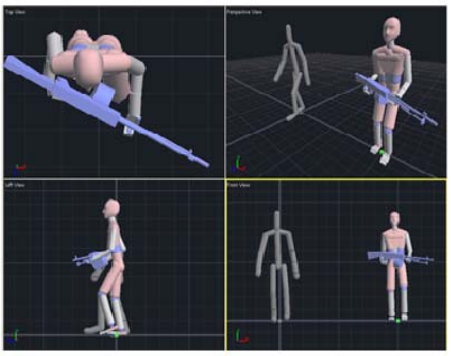
To work with multiple viewports:

1. Click the **Toggle Multiple Viewports** button  on the Main Toolbar to toggle between a single viewport and an array of four viewports. The keyboard shortcut is **V**. Currently there is no menu command to toggle between single and multiple viewports.
2. You can pan, rotate and zoom in each viewport independently. The **active viewport** is displayed with a yellow-orange border.
3. Keep in mind that some viewport settings, such as the **viewport display mode**, affect all the viewports at the same time. Likewise, entities that have been selected in one of the viewports will be displayed using the **selection colour** in all of the viewports.



Single view display

Press P for Perspective View.
Press F to toggle between Front View and Back View.
Press T to toggle between Top View and Bottom View.
Press L to toggle between Left View and Right View.



Multiple view display

Active viewport displayed with yellow border.
Press P, F, L or T to activate the corresponding viewport.
Use right-mouse context menu to change viewport type.

Multiple viewports and simulation rates

endorphin simulations typically run more slowly when you are displaying **multiple** viewports. It is good practice to toggle back to **single** viewport display before starting a new simulation. Note that **playback** rates are unaffected by the use of multiple viewports.

Perspective and orthogonal views

The standard *endorphin* viewport type is a **perspective view**. You can change the view to display one of the six **orthogonal views**. The orthogonal views are Top View, Front View, Left View, Right View, Back View and Bottom View.

You can tell which view type you are looking at by the label in the top-left corner of the viewport.

To change the viewport type:

- **Right-click** in the viewport and select **Viewport**, and then select one of the view types from the submenu. The current view type is shown with a bullet point.
- Alternatively, press one of the keyboard shortcuts:

  To change to the **Perspective View**, press **P**.

  To change to the **Top View**, press **T.** Pressing **T** again toggles between the **Top View** and the **Bottom View**.

  To change to the **Front View**, press **F.** Pressing **F** again toggles between the **Front View** and the **Back View**.

  To change to the **Left View**, press **L.** Pressing **L** again toggles between the **Left View** and the **Right View**.

To rotate an orthogonal view:

By default, endorphin has the **Ortho Lock** setting turned on. When this setting is on, you cannot rotate a viewport when it is set to one of the orthogonal view types.

1. If you want to rotate an orthogonal view, ensure **View > Ortho Lock** setting is turned off.
2. If you rotate an orthogonal view away from its usual orientation, its viewport label changes to **Orthogonal View**, with the original view type in parentheses.
3. To reset an orthogonal view back to its usual orientation, select **View > Reset Viewport**. The keyboard shortcut for Reset Viewport is **Ctrl+R**. You can also right-click and select **Reset Viewport**. Keep in mind that once the viewport orientation has been reset, it can be rotated back out of its usual orientation unless the Ortho Lock setting has been turned back on.

Saving and loading viewports

Once you have set up a viewport arrangement, you can store the arrangement into binary **viewport file**. *endorphin* viewport files have the extension **.nmv**. You can then reload the viewport arrangement in the same scene, or in other scenes. This is a useful time-saving feature

endorphin viewport files store the **viewport type**—such as Perspective or Left View, as well as the **pan**, **rotate** and **zoom** camera settings of the viewport. If you are displaying multiple viewports, the viewport type and camera settings are saved for each of the four viewports. In addition, the **Wireframe** setting is also stored for each viewport.

Most other viewport settings are **not** stored in viewport files. For example, the Ortho Lock and Camera Tracking settings, the viewport display filter and any custom viewport cameras are not stored in viewport files.

Saving and loading viewport arrangements is not available in **Character Edit Mode**.

To save the current viewport arrangement:

- Select **View > Save Viewport…** and specify a folder and filename for the viewport file.

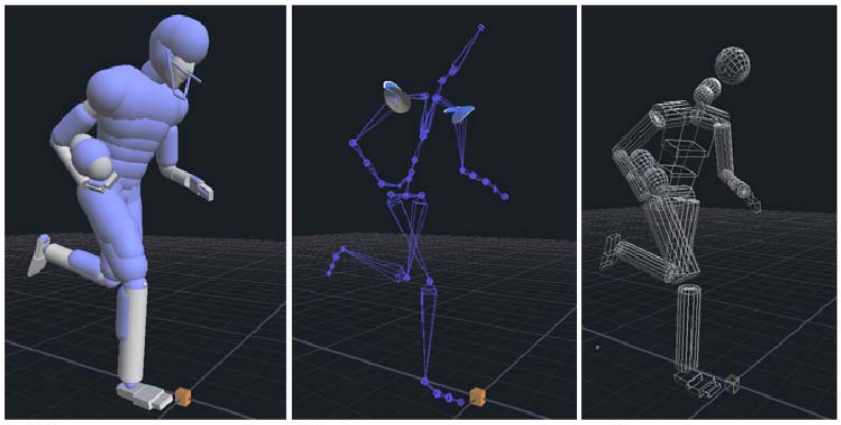
To load a different viewport arrangement:

- Select **View > Load Viewport…** and browse to select a viewport file.

Viewport display filter

endorphin scenes contain many different types of graphical entities, such as joints, mass objects, collision objects and so on. It is often useful to only show certain types of entities at a given time, depending on the particular editing task. This simplifies the viewport display, and can make editing easier. Viewports have a **display filter** which specifies which types of entities are displayed. If you are using multiple viewports, the display filter affects all the viewports.

Many entities also have their own **visibility** property, which can be toggled in the Node View. For an entity to be visible in the viewport, the entity itself must have its visibility turned on, and that entity type must also be turned on in the viewport display filter. For example, in a new scene, the Joint Limit display filter is turned on. However, you will not see any joint limits because in the default scene, the visibility for all joint limits has been turned off.



Displaying bones

endorphin joints are made up of a **connector graphic** and **a bone graphic**. You can use the display filter to show or hide the bone graphics partially independently of the connector graphics:

- Toggling the **Joints** display filter will show or hide both the connector graphics and the bone graphics.
- If the **Joints** setting is turned on, the **Bones** display filter will show or hide the bone graphics.
- If the **Joints** setting is turned off, bone graphics are not displayed, regardless of the **Bones** display filter setting.

Displaying markers

Markers are small spherical graphical items that are associated with mass objects. By default, markers are not displayed. You can show or hide markers using the viewport display filter. The motion of markers is exported when animation data is exported using the **CSM** (Character Studio Marker) file format. Keep in mind that this pipeline is not widely used and is no longer actively supported.

Changing the display filter

To change which types of entity are displayed:

- You can specify which types of entities are visible by **right-clicking** in the viewport and selecting **Display**, and then selecting or deselecting each type of viewport display item using the Display submenu. There are separate settings for mass objects, collision objects, graphical objects, bones, joints, joint limits, character cubes, markers, forces and the grid.
- Alternatively, you can adjust these settings using the View menu in the Menu Bar.

To quickly set common display filter settings:

- If you want to work with solid objects in a scene, **right-click** in the viewport and select **Shaded View**. This command displays mass objects, collision objects, graphical objects, joint limits, character cube, forces and the grid, and hides other entity types.
- If you want to work on the character bones and joints, **right-click** in the viewport and select **Skeletal View**. This command displays joints, joint limits, character cubes, forces and the grid, and hides the other entity types.
- The keyboard shortcut for toggling between Shaded View and Skeletal View is **Z**.

To set Wireframe View:

- If you want to be able to see both solid objects *and* joints, you can **right-click** in the viewport and select **Wireframe View**. This sets the display filter in the same manner as for Shaded View, except that solid objects such as mass objects, collision objects and graphical objects are rendered using a wireframe approach instead a shaded render approach. Wireframe View is useful if you have a lot of embedded mass and collision objects, and need to be able to select **through** solid objects in order to select objects embedded within them.
- To exit Wireframe View, **right-click** in the viewport and select **Shaded View** or **Skeletal View**.
- You can change the Wireframe View display settings by selecting **Options > Display** and changing settings in the Display Options dialog.

Viewport context menu

The **viewport context menu** appears when you right-click in a viewport. This menu displays many of the same commands as the **View** menu on the Menu Bar. However, using the viewport context menu can be a more efficient way of working with viewports.

Keep in mind that if you are displaying multiple viewports, some of the contect menu commands apply to the active viewport only, whereas other command apply to all the viewports.

All viewport context menu commands are also available in **Character Edit Mode**, except for the Camera Tracking toggle.

Commands that apply to the active viewport:

- **Wireframe View** toggles the wireframe mode of the active viewport.
- **Viewport…** displays a submenu that allows you to specify the camera for the active viewport. If your scene contains custom viewport cameras, these will also be displayed in the camera list.
- **Focus Viewport** rotates and pans the camera in the active viewport so that the origin—or the selected object—is displayed at the centre of the active viewport.
- **Reset Viewport** resets the camera in the active viewport back to its default pan, rotate and zoom settings.

Commands that apply globally:

- **Shaded View** displays mass and collision objects, and hides joints, in all viewports.
- **Skeletal View** displays joints, and hides mass and collision objects, in all viewports.
- **Select Character…** displays a submenu that allows you select one of the characters in the scene.
- **Show Character…** displays a submenu containing a list of the hidden characters in the scene, allowing you to redisplay one of the hidden characters.
- **HideCharacter…** displays a submenu containing a list of the visible characters in the scene, allowing you to hide one of the visible characters.
- **Display…** displays a submenu containing a list of object types, allowing you to show or hide objects of a specific type in all viewports.
- **Camera Tracking** allows you to toggle the camera tracking mode for the scene.

Viewport grid

The viewport grid is a rectangular mesh that is helpful in visualizing the ground plane. The viewport grid is turned on by default, but in some cases you might want to turn it off. For example, you may have modeled your scene with the **Use Default Ground** setting turned off, so that the viewport grid no longer reflects the location of the ground plane.

You can change the size, density and colour of the viewport grid by selecting **Options > Display** and changing settings in the **Display Options** dialog.

When you are editing a character in Character Edit Mode, *endorphin* displays the **Character Edit Mode grid**, rather than the usual **scene grid**. By default, the Character Edit Mode grid is **smaller** than the scene grid. Keep in mind that if you modify the grid size or density while in Character Edit Mode, you affect the Character Edit Mode grid, rather than the scene grid.

To show or hide the viewport grid:

- **Right-click** in the viewport and turn the **Grid** setting on or off.
- Alternatively, you can select **View > Grid** to turn the viewport grid on or off. The keyboard shortcut is **G**.

Chapter 4

System Options

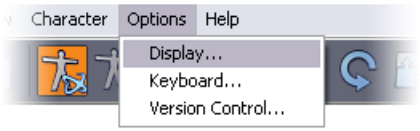
Overview

endorphin has a range of system options available.

Select **Options > Display…** to set viewport and user interface options such as colours, drawing styles and mouse input options.

Select **Options > Keyboard…** set keyboard shortcuts.

Select **Options > Version Control…** set version control settings for AlienBrain and Perforce asset version control systems.



Display options

To customise various aspects of the viewports and viewport interaction, select **Options > Display…** to display the **Display Options** dialog. The Display Options dialog contains a number of property sheets. Each property sheet contains a different group of viewport options.

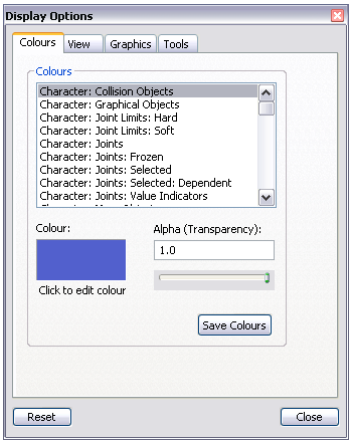
The property sheets are:

- **Colours:** Colour and transparency settings for various elements of the user interface.
- **View:** Viewport settings, such as grid settings and viewport threading settings.
- **Graphics:** Graphics settings, such as resolution and style, for various graphical elements .
- **Tools:** Interactive tool settings, such as mouse input and selection sensitivities.

Colour options

The **Colour** options allow colour settings for the user interface and viewports to be modified. Colour settings are available by selecting **Options** > **Display…** and navigating to the **Colours** property sheet.

The elements that can have their colour modified are listed in the Colours listbox. In the viewports, the main colour element groups are **Character** and **Viewport**. In the tool windows, the main colour element groups are **Property Editor**, **Node View**, **Timeline Editor** and **Event**.



Colour transparency

Certain entities in the viewport also support partial **transparency**. Drawing entities partially transparent can be useful if you want to be see other entities inside them. For example, graphical objects are displayed partially transparent by default, so that you can work with mass objects and collision objects inside the graphical object. Also, some colours such as **selection** colours, may be drawn partially transparent so that they are displayed with reduced intensity for aesthetic effect.

Standard colours

The following tables list the standard *endorphin* system colours.

Character colours

Colour	Used for
Collision Objects	Collision objects
Graphical Objects	Graphical objects
Joint Limits: Hard	Joint hard limits for skeletal and hinge joints
Joint Limits: Soft	Joint soft limits for skeletal and hinge joints
Joints	Joints
Joints: Frozen	Joints of **inactive** characters in Character Edit Mode
Joints: Selected	**Selected** joints
Joints: Selected: Dependent	Child joints of **selected** joints
Joints: Value Indicators	Joint position indicators for skeletal and hinge joints
Mass Objects	Mass objects
Markers	Markers
Objects: Frozen	Mass objects **frozen** using the Pose Move and Pose Rotate tools
Objects: Selected	**Selected** mass objects, collision objects and graphical objects
Objects: Selected: Dependent	Child collision and graphical objects of **selected** mass objects
Objects: Selected: Dynamic	**Selected** mass objects, collision objects, graphical objects and joints in **Selection** mode
Rig Node: Source	Source character rig nodes
Rig Node: Target	Target character rig nodes
Simulation Level	Mass objects, collision objects and

	graphical objects when the character **simulation mode** is No Simulation, Collision Only or Collision With Momentum
Strobed	Mass objects, collision objects and graphical objects during **strobe preview**

Event colours

Colour	**Used for**
Active Pose	Active pose event markers
Animation	Animation event markers. Also used for **viewport animation data** and motion trail display
Behaviour	Behaviour event markers
Constraint	Constraint event markers
Force	Force event markers. Also used for viewport force arrow manipulators
Motion Transfer	Motion transfer event markers
Selected	**Selected** markers
Sever	Sever event markers
Simulation	Simulation event markers
Transition	Transition event markers

Node View colours

Colour	**Used for**
Background	Background
Text	Text and connector lines

Property Editor colours

Colour	Used for
Background	Cell background
Background: Selected	Cell background of **selected** properties
Frame	Borders and cell frames
Text	Text
Text: Selected	Text of **selected** properties

Timeline Editor colours

Colour	Used for
Background	Background
Background: Selected	Background for **selected** characters
Background: Editing	Background for characters being **edited in-scene**
Dividers: Major	Horizontal character separator line
Dividers: Minor	Vertical separator line between characters and their timelines
Keyframes	Vertical keyframe lines
Loop Range	Loop range marker
Saved Frames	Save range marker
Simulated Frames	Frame buffer marker
Simulation Level	Overlay when character **simulation mode** is No Simulation, Collision Only or Collision With Momentum
Strobe	Strobe marker
Text	Text and time ruler
Text: Selected	Text for **selected** characters
Time Slider	Time slider

Viewport colours

Colour	Used for
Background	Background
Grid Lines: Major	Major grid lines
Grid Lines: Minor	Minor grid lines between major grid lines
Labels	Viewport label indicating view type and editing mode
Selected	Selected viewport border (used with multiple viewports)
Simulating	Viewport border during simulations

Editing colours

You can change the colour of many elements of the *endorphin* user interface. You can also change the transparency of various elements in the user interface.

If you change the colour of tool windows, such as the Property Editor, Node View or Timeline Editor, these changes are automatically used for other scenes. Also, changes to the viewport colours, such as grid line colours, are also automatically used in other scenes. However, changes to scene entities, such as mass objects and joints, are stored on a scene-by-scene basis.

Note Some of the Timeline Editor colour settings are not fully operational. These colour settings will be added in future versions.

To edit a colour:

1. Select **Options > Display…** to display the Display Options dialog.
2. Navigate to the **Colours** property sheet.
3. Select a colour from the **Colours** listbox.
4. Click the colour swatch to display the Colour dialog.
5. Choose a new colour and click **OK**. If you do not want to change the colour, click **Cancel**.

To edit colour transparency:

1. Select **Options > Display…** to display the Display Options dialog.
2. Navigate to the **Colours** property sheet.
3. Select a colour from the **Colours** listbox.
4. Enter a new value into the **Alpha (Transparency)** text box. The value must be between 0.0 and 1.0. You can also use the slider to set the Transparency value. A value of 0.0 is fully transparent. A value of 1.0 is fully opaque. Keep in mind that although you can modify the transparency for any colour, only a few viewport elements currently use the transparency value. You will usually modify transparency for solid viewport entities such as mass objects, collision objects, graphical objects and joint limits, and also for some modes such as **Selected** and **Frozen**.

To edit the default colour scheme for new scenes:

1. Select **Options > Display…** to display the Display Options dialog.
2. Navigate to the **Colours** property sheet.
3. Make the desired changes to colours and transparencies.
4. Click **Save Colours**. This saves your current colour and transparency settings as the default for new scenes. This affects colours for scene entities such as mass objects and joints. Other colours, such as tool window colours, are automatically used in new scenes without having to use Save Colours.

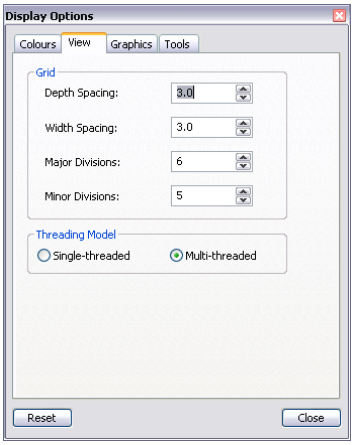
To reset the Colour options:

1. Select **Options > Display…** to display the Display Options dialog.
2. Click **Reset**. This resets the colour scheme to the *endorphin* standard colour scheme.
3. If you want to use the default colours in all scenes, click **Save Colours** after resetting the colours.

View options

The **View** options modify various aspects of the viewport appearance and operation. View options are saved globally. They are **not** saved on a scene-by-scene basis. View settings are available by selecting **Options > Display…** and navigating to the **View** property sheet.

The **Reset** button does not modify any of the View options. For the Colours, Graphics and Tools property sheets, the Reset button will reset the settings back to the *endorphin* default settings.



To change the viewport grid size:

1. Select **Options > Display…** to display the Display Options dialog.
2. Navigate to the **View** property sheet.
3. To change the spacing between each grid line, modify the **Depth Spacing** and **Width Spacing** values.
4. To change the extent of the grid, modify the **Major Divisions** value.
5. To change the number of subdivision lines between each major division line, modify the **Minor Divisions** value.

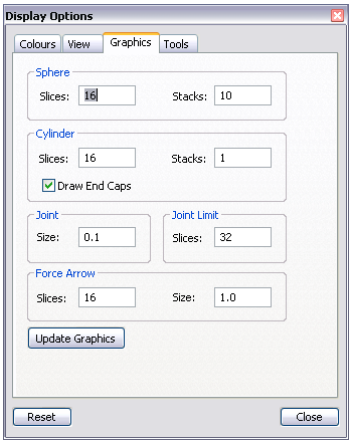
To change the viewport threading model:

1. Select **Options > Display…** to display the Display Options dialog.
2. Navigate to the **View** property sheet.
3. Change the **Threading Model** setting to Single-Threaded or Multi-Threaded. You can use this setting to speed up *endorphin* on computers that have multiple processors. The default setting is Multi-Threaded. However, on certain

configurations of hardware, the Multi-Threaded setting may cause problems. If you find that *endorphin* is not running correctly, or appears to freeze at certain times, try changing the Threading Model setting to Single-Threaded. This problem has been documented for some ATI graphics cards. For more information, please consult the *endorphin* User Community Forum.

Graphics options

The **Graphics** options modify various aspects of the graphics primitives display. Graphics options are saved globally. They are **not** saved on a scene-by-scene basis. Graphics settings are available by selecting **Options > Display…** and navigating to the **Graphics** property sheet.



To change the way graphics primitives are drawn:

1. Select **Options > Display…** to display the Display Options dialog.
2. Navigate to the **Graphics** property sheet.
3. To change the way spheres are drawn, modify the **Slices** and **Stacks** settings in the Sphere group. Use higher settings for smoother shapes, and use lower settings for faster redraws. These settings affect spherical mass objects and collision objects in Shaded or Wireframe modes. The **Stacks** value will also have an effect on the end caps of sphyl objects
4. To change the way cylinders are drawn, modify the **Slices** and **Stacks** settings in the Cylinder group. Use higher settings for smoother shapes, and use lower settings for faster redraws. These settings affect cylindrical mass objects and collision objects in Shaded or Wireframe modes. The Draw End Caps setting is currently not used.

5. To change the way joints are drawn, modify the **Size** setting in the Joint group. The size setting affects the way the joints are drawn—by increasing their radius—rather than the bones between the joints.
6. To change the way joint limits are drawn, modify the **Slices** setting in the Joint Limit group. Use higher settings for smoother joint limit graphics, and use lower settings for faster redraws.
7. To change the way force arrows are drawn, modify the **Slices** setting in the Force Arrow group. Use higher settings for smoother force arrow graphics, and use lower settings for faster redraws. Modify the **Size** setting to change the size of force arrows. Keep in mind that the size of force arrows is also dependent upon their Strength property.
8. The changes only come into effect when you click **OK**. If you want to see effect of your changes before you click OK, click **Update Graphics**.

To reset the Graphics options:

1. Select **Options > Display…** to display the Display Options dialog.
2. Navigate to the **Graphics** property sheet.
3. Click **Reset**. This resets all the settings on the Graphics property sheet back to the *endorphin* defaults.

Tools options

The **Tools** options modify various aspects of the viewport interaction. Tools options are saved globally. They are **not** saved on a scene-by-scene basis. Tools settings are available by selecting **Options > Display…** and navigating to the **Tools** property sheet.

To change way Move, Rotate and Rotate tools are drawn:

1. Select **Options > Display…** to display the Display Options dialog.
2. Navigate to the **Tools** property sheet.
3. To change the line thicknesses of the tool, modify the **Line Width** setting.
4. To change the size of the tool, modify the **Scale** setting.

To change pan, rotate and zoom operate:

1. Select **Options > Display…** to display the Display Options dialog.
2. Navigate to the **Tools** property sheet.
3. To change the sensitivity of mouse rotation, modify the **Rotating** setting. Enter larger values to rotate rapidly. Enter smaller values to rotate slowly.
4. To change the sensitivity of mouse panning, modify the **Panning** setting. Enter larger values to pan rapidly. Enter smaller values to pan slowly.
5. To change the sensitivity of mouse zooming, modify the **Mouse Zoom** setting. Enter larger values to zoom rapidly. Enter smaller values to zoom slowly
6. To change the sensitivity of mouse wheel zooming, modify the **Wheel Zoom** setting. Enter larger values to zoom rapidly. Enter smaller values to zoom slowly.

To change pan, rotate and zoom operate:

1. Select **Options > Display…** to display the Display Options dialog.
2. Navigate to the **Tools** property sheet.
3. To change the sensitivity of selection, modify the **Sensitivity** setting. When you click in the viewport, you do not have to select an entity directly. The Sensitivity setting measures the number of pixels away from the entity that you can click and still select the entity. Keep in mind that if the Sensitivity setting is too large, many potential target entities may be present in the selection zone around a mouse click, and you might encounter unexpected entity selection.

To reset the Tools options:

1. Select **Options > Display…** to display the Display Options dialog.
2. Navigate to the **Graphics** property sheet.
3. Click **Reset**. This resets all the settings on the Tools property sheet back to the *endorphin* defaults.

Keyboard options

To use *endorphin* more efficiently, a number of commands have been associated with **keyboard shortcuts**. If a command has a corresponding keyboard shortcut, the shortcut appears alongside the command name in the Menu Bar.

Not every command is available from the Menu Bar. Commands available from context menus do not have the corresponding keyboard shortcut displayed in the context menu, so it may not be obvious that these commands also have keyboard shortcuts.

To see the full list of commands that have been associated with keyboard shortcuts, select **Options > Keyboard…** to display the **Keyboard Options** dialog.

In this User Guide, we always refer to the default keyboard shortcuts in the **endorphin Standard** keyboard profile. However, you can change the active profile and define your own keyboard shortcuts.

Standard keyboard shortcuts

The following tables list the default *endorphin* keyboard shortcuts. In this User Guide, we always refer to the default keyboard shortcuts in the **endorphin Standard** keyboard profile. However, you can change the active profile and define your own keyboard shortcuts.

Application commands

Command	Keyboard shortcut
New Scene…	Ctrl+N
Open Scene…	Ctrl+O
Save Scene…	Ctrl+S
Full Screen	Ctrl+Shift+F
Lock Layout	Ctrl+Shift+L
Toggle Node View	N
Toggle Motion Transfer Editor	M
Help	F1

Viewport commands

Command	Keyboard shortcut
Zoom In	Plus (+)
Zoom Out	Minus (-)
Toggle Grid	G
Reset Viewport	Ctrl+R
Focus Viewport	Ctrl+F
Toggle Camera Tracking	C
Toggle Display Mode	Z
Toggle Multiple Viewports	V
Toggle Strobing	Ctrl+Shift+S
Perspective View	P
Top/Bottom View	T

Left/Right View	L
Front/Back View	F
Custom View 1	1
Custom View 2	2
Custom View 3	3
Custom View 4	4

Editing commands

Command	**Keyboard shortcut**
Cut	Ctrl+X
Copy	Ctrl+C
Paste	Ctrl+V
Delete	Delete
Undo	Ctrl+Z
Redo	Ctrl+Y
Selection Tool	R
Move Tool	Q
Rotate Tool	W
Scale Tool	E
Pose Move	A
Pose Rotate	S
Toggle Coordinate System	X
Create Key	B

Simulation and playback commands

Command	**Keyboard shortcut**
Simulate	Ctrl+Spacebar
Play/Stop	Spacebar
Play Backwards	Alt+Spacebar

Go To Previous Frame	LeftArrow
Go To Next Frame	RightArrow
Go To Start Frame	Ctrl+LeftArrow
Go To End Frame	Ctrl+RightArrow

Dialog commands

Command	Keyboard shortcut
Import…	I
Export…	Ctrl+E
Reload Character…	Ctrl+L

Character Edit Mode commands

Command	Keyboard shortcut
Toggle Active Character	Ctrl+Tab
Mirror Left To Right	Alt+LeftArrow
Mirror Right To Left	Alt+RightArrow
Mirror Selected Left To Right	Ctrl+Alt+Left
Mirror Selected Right To Left	Ctrl+Alt+Right

Editing keyboard shortcuts

endorphin has a number of **keyboard shortcut profiles**. Each profile defines a collection of keyboard shortcuts. The shortcut profiles are listed at the top of the Keyboard Options dialog.

The default shortcut profile is called **endorphin Standard**. You cannot edit this profile.

The next shortcut profile is called **endorphin User-Defined**. You can edit this profile. By default this profile has the same keyboard shortcuts as the endorphin Standard group.

The other profiles are called **3ds max**, **Maya** and **SOFTIMAGE|XSI**. All these profiles may be edited. These profiles have similar keyboard shortcuts to the endorphin Standard profile, but in some cases the shortcuts have been changed to better fit the defaults found in 3ds max, Maya and SOFTIMAGE|XSI respectively.

To change the active keyboard shortcut profile:

1. Select **Options > Keyboard…** to display the Keyboard Options dialog.
2. Select the desired keyboard shortcut profile.
3. Click **OK**.

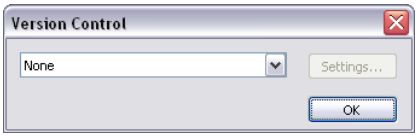
To edit a keyboard shortcut profile:

1. Select **Options > Keyboard…** to display the Keyboard Options dialog.
2. Select the desired keyboard shortcut profile. Keep in mind that the endorphin Standard profile cannot be edited.
3. Select a specific command from the list of commands.
4. Click **Edit Hotkey…** to display the **Edit Hotkey** dialog, or **double-click** on the command name.
5. Select the text control.
6. Press a new key, or combination of keys. Most keys are available for use as a keyboard shortcut. You cannot use a key combination that is already in use in the edited profile. If you attempt to do so, *endorphin* will display an error message.
7. Click **OK** to accept this change, or **Cancel** to cancel this change.

Version control options

endorphin includes integrated support for the Perforce® and Avid Alienbrain® **version management systems**. When version control has been enabled, endorphin performs version control operations whenever documents are loaded and saved, and also whenever data is imported or exported.

To use version control with *endorphin*, you must have already installed Perforce or Avid Alienbrain. With a properly configured client, and the corresponding version control options set, *endorphin* will perform version control when it works with files that are included in your **managed workspace**.



To change version control settings:

1. Select **Options > Version Control…** to display the **Version Control** dialog.
2. If you have installed Perforce or Avid AlienBrain, these will appear as items in the version control list. Select your version control system that you use from the drop-down list.
3. Click the **Settings…** button to configure the interaction between *endorphin* and the version control system. You can specify the appropriate behaviour for the following file events: opening files, closing files, saving to replace existing files and saving to create new files. For each event type, you can specify whether *endorphin* should **always** perform the appropriate version control operation, **never** perform the operation, or **prompt** the user at each file event

    - The default behaviour is when opening files is Ask Before Checking Out. The alternative options are Always Check Out and Never Check Out.
    - The default behaviour is when closing files or saving to replace existing files is Ask Before Checking In. The alternative options are Always Check In and Never Check In.
    - The default behaviour when saving to create new files is Ask Before Adding To Database. The alternative options are Always Add To Database and Never Add To Database.

    You should generally leave version control events set to **prompt** until you are comfortable with how *endorphin* interacts with your versioning environment.

Chapter 5

Scenes

What is an *endorphin* scene?

The main *endorphin* document type is a **scene**.

Scenes contain the definition of a **virtual world** that is populated by **characters**. One character always represents the **environment**. Other characters may represent humans or humanoids, or may represent animals, vehicles or other types of inanimate objects.

Scenes also contain a set of time-based **events**. These allow you to modify a simulation by applying forces, constraints, poses and behaviours to characters.

Scenes are saved as **.ens** binary files. The virtual world, its characters and any time-based events are all stored in scene files. However, any simulated frames are **not** stored in scene files. This is because they can be readily reproduced by reopening the scene and resimulating.

Creating new scenes

Scenes are the main *endorphin* document type. It is easy to create, open and save scenes.

To create a new scene:

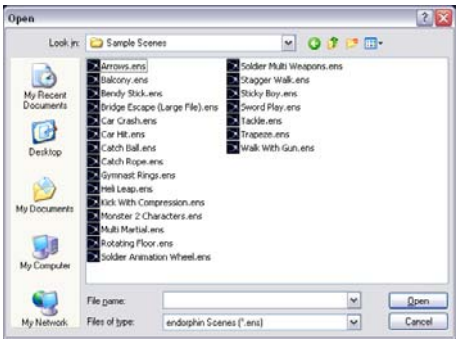
- Select **File > New Scene** The keyboard shortcut is **Ctrl+N**. When you create a new scene, the viewport camera angle is reset. By default, new scenes contain an empty **Environment** character, and a single instance of the standard simulation character called **Character01**, arranged at the centre of the virtual world in a T-pose. If you have an existing scene already open, *endorphin* will prompt you to save any changes before creating the new scene.

Opening existing scenes

Scenes are the main *endorphin* document type. It is easy to create, open and save scenes.

To open an existing scene:

Select **File > Open Scene…**. The keyboard shortcut is **Ctrl+O**. The **Open** dialog will be displayed, so you can browse to the **.ens** file you want to open. If you have an existing scene already open, *endorphin* will prompt you to save any changes before displaying the Open dialog.



To open a recently-used scene:

Each time you work with a scene, it is added to the **Most Recently Used** list. You can select **File** and choose one of the scenes from the Most Recently Used list to quickly re-open one of these scenes without the need to use the Open dialog. Up to ten scenes appear in the Most Recently Used list. This list is saved by *endorphin* and is available the next time you run the application.

To open a scene in a new instance of *endorphin*:

You can double-click on an **.ens** scene file in the Windows File Explorer in order to launch a new instance of *endorphin* with that scene preloaded.

Learning Edition scenes

Keep in mind that scenes created in *endorphin* Learning Edition cannot be opened in *endorphin* or *endorphin* Educational Edition, and vice versa.

Saving scenes

Scenes are the main *endorphin* document type. It is easy to create, open and save scenes.

To save a scene:

- Select **File > Save Scene**. The keyboard shortcut is **Ctrl+S**. Alternatively, you can click the **Save Scene** button in the Main Toolbar.
- If you have not already saved the scene, the **Save As** dialog will be displayed, so you can browse to a folder in which to save the scene. Alternatively, you can browse to an existing scene file if you want to replace that scene. If you browse to an existing scene, a warning dialog is displayed so that you can confirm that you want to replace the existing scene.

- If you have already saved the scene, *endorphin* simply replaces saves the scene to its current save location.

To save a scene to a new location:

- Select **File > Save Scene As…**. This will display the **Save As** dialog, allowing you to specify a new location in which to save the scene. You can also select an existing scene if you want to replace that scene.

The virtual world, its characters and any time-based events are all stored in scene files. However, any simulated frames are **not** stored in scene files. This is because they can be readily reproduced by reopening the scene and resimulating.

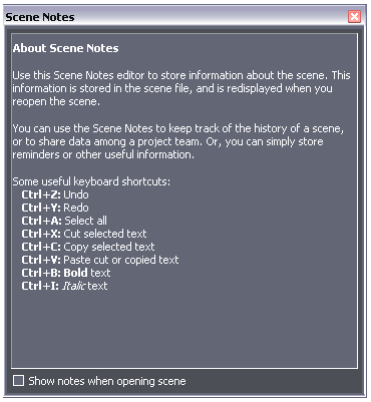
Closing scenes

There is no specific command to close a scene. *endorphin* is a **single-document interface** system, which means that you can edit one scene at a time.

If you create a new scene, or open another existing scene, or close the application itself, then the scene currently being edited is closed automatically. If you have made any unsaved changes to the scene, a **Save** dialog appears. This dialog gives you the opportunity of saving any changes you have made.

Scene notes

You can add **notes** to a scene. This is a useful way of keeping track of changes to the scene, or to share information with other members of a team working on the same scene.



To edit the Scene Notes:

1. Select **View > Scene Notes** to display the **Scene Notes** editor.
2. Enter text directly into the text box. You can also copy-and-paste text from other word processing applications such as Microsoft Word.
3. If you want to use different fonts, font sizes or font colours, you will need to format the text in a word processing application such as Microsoft Word, and then copy-and-paste the formatted text into the Scene Notes. Once you have done this, any additional text that you enter directly into the Scene Notes will use the formatting that you have pasted. Keep in mind that you can change the **bold** and *italic* settings of Scene Notes text directly within the Scene Notes editor itself.

To show the Scene Notes editor automatically:

If you want to show the Scene Notes editor automatically each time you open the scene, select the **Show notes when opening scene** option in the Scene Notes editor. This setting is stored on a per-scene basis. Most of the sample scenes that are shipped with *endorphin* have this option turned on.

To cut, copy and paste text:

- Use **Ctrl+C** to copy the selected text to the clipboard.
- Use **Ctrl+X** to cut the selected text to the clipboard.
- Use **Delete** to cut the selected text **without** placing it on the clipboard.
- Use **Ctrl+V** to paste text from the clipboard.

To undo and redo changes:

- Use **Ctrl+Z** to undo changes.
- Use **Ctrl+Y** to redo changes.

To format text:

- Use **Ctrl+B** to toggle the selected text between regular and bold. Note that Ctrl+B does not operate if the selected text contains a mix of regular and bold text.
- Use **Ctrl+I** to toggle the selected text between regular and italic. Note that Ctrl+I does not operate if the selected text contains a mix of regular and italic text.
- When you enter text, its initial font, font size, color, bold and italic settings are based on the prevailing setting at the cursor position. This is a familiar convention used by many word processing systems such as Microsoft Word.

To navigate:

- Use **Home** to move the cursor to the start of a line.
- Use **End** to move the cursor to the end of a line.
- Use **Ctrl+Home** to move the cursor to the start of the Scene Notes.
- Use **Ctrl+End** to move the cursor to the end of the Scene Notes.
- Use **PgUp** to move the cursor up by one page.
- Use **PgDn** to move the cursor down by one page.
- Use the **UpArrow** and **DownArrow** keys to move up and down by one line.
- Use the **LeftArrow** and **RightArrow** keys to move left and right by one character.
- Use **Ctrl+UpArrow** and **Ctrl+DownArrow** keys to move up and down by one paragraph.
- Use **Ctrl+LeftArrow** and **Ctrl+RightArrow** keys to move left and right by one word.

To select text:

- Hold down **Shift** while using any of the navigation keys to select the text between the initial cursor position and the new cursor position.
- Use **Ctrl+A** to select all the text.
- **Double-click** to select the current **word** of text.
- **Triple-click** to select the current **paragraph** of text.

To set text bulleted or numbered:

- Use **Ctrl+Shift+L** to toggle the selected text between bulleting, numbering, Roman numbering and none.

To set text alignment:

- Use **Ctrl+L** to align the current paragraph **Left**.
- Use **Ctrl+R** to align the current paragraph **Right**.
- Use **Ctrl+E** to align the current paragraph **Centre**.

To set line spacing:

- Use **Ctrl+1** to set the line spacing to **Single Spacing** for the current paragraph. This is the default spacing.
- Use **Ctrl+5** to set the line spacing to **1.5 Line Spacing** for the current paragraph.
- Use **Ctrl+2** to set the line spacing to **Double Spacing** for the current paragraph.

To create accented characters:

- Use **Ctrl+'** followed by the appropriate letter, to create an accent **acute (é)**.
- Use **Ctrl+Shift+6** followed by the appropriate letter, to create an accent **circumflex (ê)**.
- Use **Ctrl+`** followed by the appropriate letter, to create an accent **grave (è)**.
- Use **Ctrl+~** followed by the appropriate letter, to create an accent **tilde (ñ)**.
- Use **Ctrl+;** followed by the appropriate letter, to create an accent **umlaut (ö)**.
- Use **Ctrl+,** followed by the appropriate letter, to create an accent **cedilla (ç)**.

Environment character

Scenes always contain a character called **Environment**. This environment character contains mass objects and collision objects that represent the physical world in which the other characters in the scene coexist.

You will typically populate the environment character with entities that represent rooms, buildings and other objects in the built environment. You may also add entities representing terrain and other objects in the natural environment.

Environment ground plane

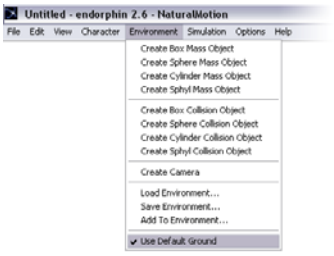
The virtual world contains a **ground plane**. The ground plane is a large collision object used to represent the a flat surface that is coincident with the viewport grid. The ground plane is a special collision object that is not visualised in viewports.

You can turn off the ground plane. This does not delete the ground plane from the virtual world. Instead, it disables the ground plane collision object. When the ground plane is turned off, simulated objects can fall below the viewport grid plane. This can be useful if your scene does not have surface that corresponds to the default ground plane.

The ground plane is turned on by default.

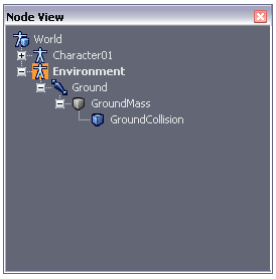
To turn the ground plane off:

Select **Environment > Use Default Ground** and ensure that it is unchecked.



Editing the environment

The environment character is called **Environment**. It contains a skeleton root joint called **Ground**. The skeleton root joint has a mass object called **GroundMass**, which has a corresponding collision object called **GroundCollision**. This object represents the **ground plane**.



Common tasks

- If you want to add objects that **cannot be moved** by other characters, add collision objects directly to the environment character. This approach is useful for modeling objects like walls.
- If you want to add objects that **can be moved** by other characters, or by forces such as gravity, add one mass object to the environment for each of these objects. You can then select these mass objects and create child collisions objects to define the shape of these objects. This approach is useful for modeling objects like tables and chairs.

To edit the environment ground plane:

- You can move, resize or even delete the **GroundMass** and **GroundCollision** objects. However, this is generally not recommended.
- If you do not require the ground plane in your scene, turn off the ground plane by selecting **Environment > Use Default Ground** and ensuring that it is unchecked. This does not delete the GroundMass or GroundCollision objects, but it does ensure that they play no role in simulations.

Adding objects to the environment

To add mass objects to the environment:

1. Press **Esc** to ensure that no entities in the scene are selected.
2. Select **Environment > Create Box Mass Object** to create a box mass object. Similar commands exist to create cylinder, sphere or sphyl mass objects.
3. Alternatively, use the Main Toolbar buttons to create mass objects. Click the **Create Box Mass Object** to create a box mass object. Adjacent buttons exist to create sphere, cylinder and sphyl mass objects.

Note When mass objects are added to the environment character, they have an **automatic collision object** included in their definition. This collision object can be edited when the mass object is edited, but cannot be deleted. Automatic collision objects are a special feature of environment characters.

To add collision objects to the environment:

1. Press **Esc** to ensure that no entities in the scene are selected. This ensures that any new collision objects are created as child objects of the **GroundMass** mass object.
2. Alternatively, if you have created some additional mass objects in the environment, and you want to create a new collision object for one of these mass objects, use the viewports to select one of the mass objects in the environment character.
3. Select **Environment > Create Box Collision Object** to create a box collision object. Similar commands exist to create cylinder, sphere or sphyl collision objects.
4. Alternatively, use the Main Toolbar buttons to create collision objects. Click the **Create Box Collision Object** to create a box collision object. Adjacent buttons exist to create sphere, cylinder and sphyl collision objects.

Removing objects from the environment

To delete mass objects from the environment:

1. Use the viewports to select the mass objects that you want to delete.
2. Select **Edit > Delete** to delete the objects. Alternatively, press **Delete**.

Note It is recommended that you do not delete the **GroundMass** mass object.

To delete collision objects from the environment:

1. Use the viewports to select the collision objects that you want to delete.
2. Select **Edit > Delete** to delete the objects. Alternatively, press **Delete**.

Note It is recommended that you do not delete the **GroundCollision** collision object.

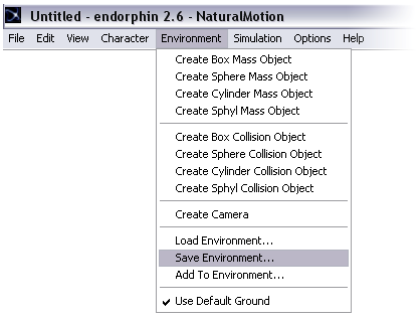
Saving and loading environments

After you have edited the environment, you may want to **reuse** the environment, or parts of the environment, in other scenes. *endorphin* allows you to save the environment to an external file, and load this environment file in other scenes.

Alternatively, you can generate environment files using scripts in other animation systems, such as Maya and 3DStudioMAX, and load these into your scene.

Environments are stored in text-based character files with **.nmc** extensions.

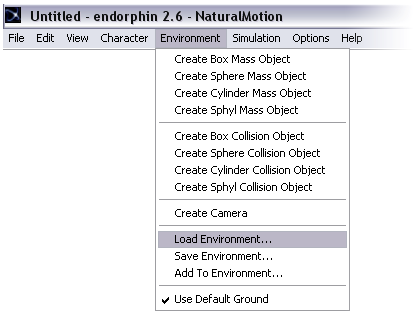
To save the environment:



1. Select **Environment > Save Environment...**.
2. The **Save As** dialog appears. Browse to a folder in which to save your environment character, or select an existing **.nmc** character file to overwrite.

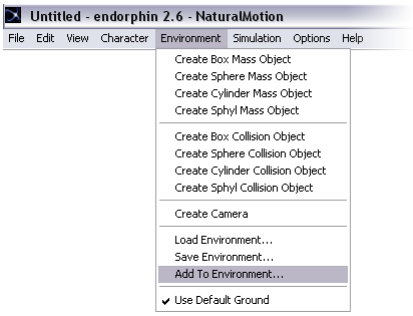
3. Click **Save** to save the environment character to an **.nmc** text file. Unlike other character files, environment character files are text files rather than binary files. This is so that they can be easily generated by scripts in other animation systems.

To load an environment:



1. Select **Environment > Load Environment…**.
2. The **Open** dialog appears. Browse to an **.nmc** character file that contains the environment character that you want to use.
3. Click **Open** to load this environment. This **deletes** any existing mass and collision objects in your environment character, and **replaces** them with the environment character that you have selected. This process cannot be undone, so use care when loading environment files.

To add to the environment:



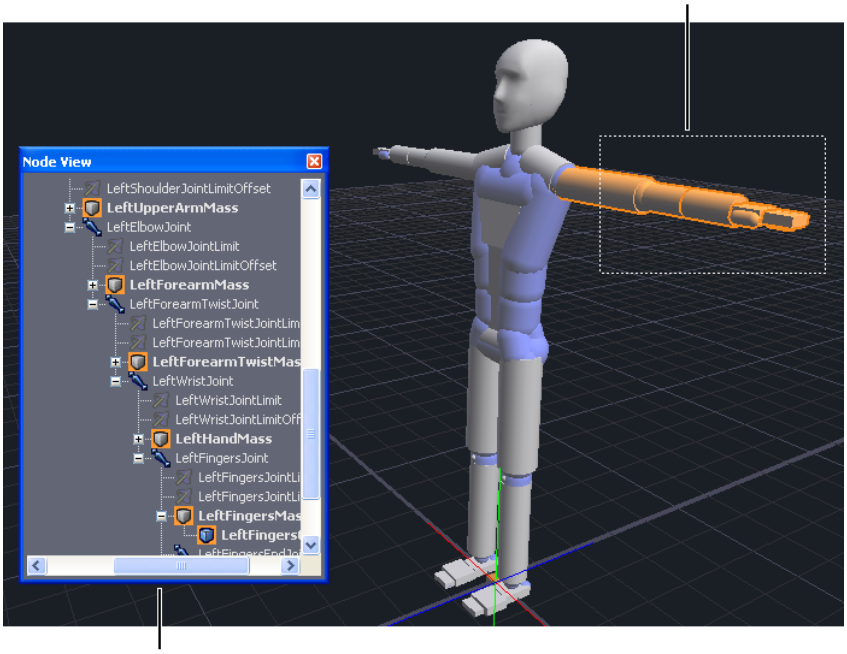
1. Select **Environment > Add To Environment…**.
2. The **Open** dialog appears. Browse to an **.nmc** character file that contains the environment character that you want to use.
3. Click **Open** to add this environment to your scene. This **merges** the mass objects and collision objects in the selected environment file to your scene environment. No existing mass or collision objects are deleted.

Selecting objects

You interact with objects in the scene by first **selecting** them. Once an object has been selected, you can inspect and modify its properties using the Property Editor. You can also use various viewport-based tools to move, rotate and scale many objects.

When objects are selected, they are displayed in viewports using the **selection colour**. In addition, they are also displayed in the Node View using the selection colour. Selected characters are also displayed using the selection colour in the Timeline Editor. By default, the selection colour is **orange**.

In order to select entities using viewports, you must first ensure that the **Select** tool is active. By default, the Select tool will always be active, unless you have activated one of the editing tools, such as the Move or Rotate tools.



Working with selection

To activate the Select tool:

- To activate the Select tool, select **Edit > Select**. The keyboard shortcut is **R**.

  Alternatively, you can click the **Select** button  in the Main Toolbar. When the Select tool is active, it will be displayed in the Main Toolbar using the **selection colour**.

To select objects individually:

1. Ensure that the **Select** tool is active.
2. To select a single object, **click** on some part of the entity in the viewport. All other objects in the scene will be deselected.

   You do not need to click directly on an entity to select it. If you click within the **Selection Sensitivity** threshold of an entity, you can also select it. The default selection sensitivity is 10 pixels. Selection sensitivity is useful when selecting items such as a joints, that are made up of long, thin line elements that might be awkward to select otherwise.

   To change the selection sensitivity, select **Options > Display… > Tools**, and enter a value into the **Sensitivity** text box.

To select objects behind other objects:

1. Ensure that the **Select** tool is active.
2. To select a single object, **click** on some part of that object in the viewport. All other objects in the scene will be deselected. The object **nearest** to the screen plane will be selected.
3. To select the object **behind** the nearest object, click **again** at exactly the same position. If there are no objects behind the nearest object, it will remain selected.
4. Continue clicking at the same point in the viewport to progressively select objects further and further from the screen plane. When you reach the last object, the next click will select the nearest object. The process repeats in a **cyclic** manner.

To box-select objects:

1. Ensure that the **Select** tool is active.
2. To box-select objects, **click** in the viewport and **drag**. When you release the mouse, any entity that is fully enclosed or partially enclosed by the selection rectangle will be selected. This includes objects that have been obscured by other objects. All other objects in the scene will be deselected.

To additively select objects:

1. Ensure that the **Select** tool is active.
2. To additively select objects, hold down the **Shift** key and select or box select entities using any of the viewports. If you select an object that is already selected, it will remain selected.

To toggle object selection:

1. Ensure that the **Select** tool is active.
2. To toggle object selection, hold down the **Ctrl** key and select or box select entities using any of the viewports. If you select an object that was not previously selected, it will be selected. If you select an object that was previously selected, it will be deselected.

Problems selecting joint limits

You should be able to select **any** object that appears in the viewport—with the exception of frozen graphical objects and inactive characters in Character Edit Mode. However, some users may find problems selecting **joint limits** using the viewport. If you are having problems selecting joint limits in the viewport, you will need to use the **Node View** to select joint limits instead. This issue may be addressed in future releases of *endorphin*.

Dependent selections

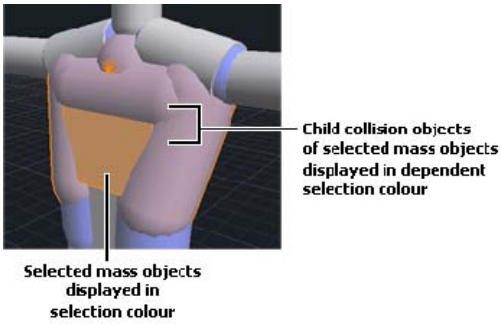
When you select an entity, it is displayed in the **selection colour** in the viewports and in the Node View. If the entity is a character, it is also displayed in the selection colour in the Timeline Editor. By default, the selection colour is orange.

Many entities have a corresponding set of entities called **dependent entities**. When an entity is selected, its dependent entities are displayed in the viewports using the **dependent selection colour**. By default, the dependent selection colour is dark orange.

Dependent selection for mass objects

When mass objects are selected, their immediate child collision objects and immediate child graphical objects are displayed as the dependent selection.

Visualising the dependent selection of a mass object is useful. It allows you to quickly identify the cluster of collision objects that define the collision surface of the mass object.

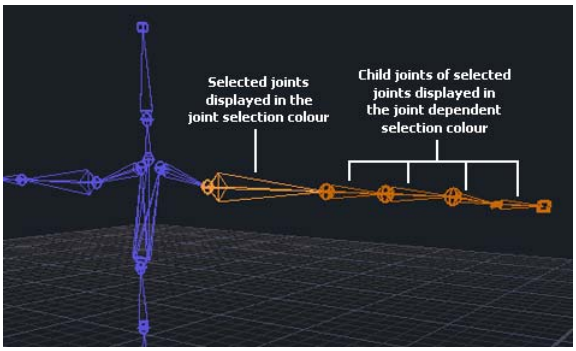


When a mass object is selected which has dependent selections, using the **Move** and **Rotate** tools will also affect these dependent selections. They will maintain their relative position to the parent mass object but will move with it. Using the **Scale** in such a case however does not effect dependent selections. Their size will remain unchanged if the parent mass object is scaled up or down.

Dependent selection for joints

When joints are selected, any child joints, child mass objects, child collision objects or child graphical objects are displayed using the dependent selection colour. This applies all the way down the joint hierarchy until the end joints are reached.

Visualising the dependent selection of a joint is useful. It allows you to quickly identify the parts of the character that will be affected when a joint is moved or rotated.



Dependent selection for timeline events

When timeline events are selected, entities that the event applies to are displayed using the dependent selection colour.

- For **active pose**, **animation** and **transition events**, the event applies to **joints**. Selecting any of these events will display the target joints in the dependent selection colour.
- For **behaviour**, **constraint**, **force** and **sever events**, the event applies to **mass objects**. Selecting any of these events will display the target mass objects in the dependent selection colour.
- For **motion transfer** and **simulation events**, the event applies to **characters**. Selecting any of these events will display the target character in the dependent selection colour.

Visualising the dependent selection of a timeline event is useful. It allows you to quickly identify the parts of the character, or the characters themselves, that will be affected when the event is triggered.

Local coordinate systems

The *endorphin* scene has a **global** coordinate system. In addition, every object also has a **local coordinate system**.

Using local coordinate systems

- When you are using the **Move**, **Rotate**, **Pose Move** or **Pose Rotate** tools to manipulate objects, you can choose whether you want to move and rotate in the **global** scene coordinate system, or in the **local** coordinate system of the selected object.
- When you are using the **Scale** tool, you can only work in the **local** coordinate system of the selected object.
- If you have multiple objects selected, the local coordinate system used by these tools is based on the **first** selected object.
- When using **local** coordinate systems, the tool manipulator graphic for the active tool will display its **red**, **green** and **blue** arrows along the local **X**, **Y** and **Z** axes, respectively.
- As a general rule, the local **X** axis for **joints** runs axially along the bone.
- As a general rule, the local **Z** axis for **cylindrical** and **sphyl** mass and collision objects runs along their axis.

To toggle the active coordinate system:

- Select **Edit > Toggle Coordinate System**. The keyboard shortcut is **X**. If you have the Move, Rotate, Pose Move or Pose Rotate tools active, the tool manipulator graphic will update to reflect the active coordinate system.

Changing the active editing tool

When you are editing a scene, you will often be using different editing tools, such as Select, Move, Rotate and Scale. There are a number of ways to quickly change the active tool.

To change the active tool in the viewport:

- **Double-click** on the Move, Rotate or Scale tool manipulators to change the active editing tool. If you double-click on the Move tool manipulator, the Rotate tool is activated. If you double-click on the Rotate tool manipulator, the Scale tool is activated. If you double-click on the Scale tool manipulator, the Move tool is activated.

To change the active tool using keyboard shortcuts:

- The keyboard shortcuts for the Move, Rotate, Scale and Select tools are **Q**, **W**, **E** and **R** respectively. These have been intentionally designed so that they are adjacent keys on most keyboards. You can edit keyboard shortcuts by selecting **Options > Keyboard…**.

Cutting and pasting

The viewports provide support for editing operations, such as **cutting**, **copying**, **pasting** and **deleting** entities. These operations main apply to objects such as mass objects and collision objects. You cannot perform these operations on some objects such as joints or joint limits.

- **Copying a selected entity** copies the entity into the Windows clipboard so that it can be **pasted** elsewhere.
- **Cutting a selected entity** will delete the entity. It also copies the entity into the Windows clipboard so that it can be **pasted** elsewhere.
- **Pasting** is the process of creating a new entity using the contents of the Windows clipboard.
- **Deleting a selected entity** will delete the entity, without making a copy.

To cut the selected entities:

1. Select the entities to be cut. Only mass objects and collision objects can be cut.
2. Select **Edit > Cut**. The keyboard shortcut is **Ctrl+X**. The selected objects are deleted from the scene and copied into the Windows clipboard.

To copy the selected entities:

1. Select the entities to be copied. Only mass objects and collision objects can be copied.
2. Select **Edit > Copy**. The keyboard shortcut is **Ctrl+C**. The selected objects are copied into the Windows clipboard.

To paste from the clipboard:

1. Select **Edit > Paste**. The keyboard shortcut is **Ctrl+V**.
2. If any mass objects or collision objects have been **cut** or **copied** to the Windows clipboard, then new corresponding mass objects and collision objects will be created and added to the scene.
3. If you are pasting **collision objects**, the pasting behaviour depends on whether or not any other mass objects in the scene are directly or indirectly selected. (If a collision object is selected, its parent mass object is indirectly selected; similarly, if a joint is selected, its child mass object is indirectly selected.)
    - If no mass objects are selected, the collision objects are pasted as children of the default **GroundMass** mass object of the **Environment** character.
    - If one or more mass objects in the Environment are directly or indirectly selected, a separate set of collision objects are pasted as children of each of the selected mass objects.
    - If you are in **Character Edit Mode**, you must have at least one mass object directly or indirectly selected—otherwise it is not possible to create new collision objects by pasting, since no target mass object has been specified. Again, if **multiple** mass objects are selected, a separate set of collision objects are pasted for each selected mass object.

      A common requirement in Character Edit Mode is to **fill out** an OBJ surface mesh with collision objects. This can require creating a large number of collision objects. You can quickly create new collision objects by selecting an existing collision object and copying-and-pasting it, keeping the original copied collision object selected. This process creates a set of **sibling** collision objects that all share the same parent mass object as the original copied collision object.

Note If you are **copying-and-pasting**, the pasted objects will be created at the same positions as the copied objects. This can be slightly confusing, as it may appear as though no additional objects have been created.

To delete the selected entities:

1. Select the entities to be deleted. Only mass objects, collision objects and characters can be deleted.
2. Select **Delete**. The keyboard shortcut is **Delete**. The selected objects are deleted from the scene.

Undo and redo

The editing system allows you to **undo** and **redo** some selection and editing operations. This is a useful way to quickly revert recent changes that you have made. Each time you make a change to a scene, that change is stored as a new undo item in the **undo history**.

Not all operations have undo and redo support. Also, some operations, such as changing the active tool, clear the undo history. It is good practice to save different versions of a scene while you are editing it, so that you can revert to an earlier version of the scene if required. You should not rely entirely on the undo system as a way of keeping track of changes made to a scene.

The undo history is cleared when the scene is closed.

To undo an operation:

- Select **Edit > Undo**. The keyboard shortcut is **Ctrl+Z**. Alternatively, click the **Undo** button  on the Main Toolbar. You can undo multiple times until there are no more undo items in the undo history.

To redo an operation:

- Select **Edit > Redo**. The keyboard shortcut is **Ctrl+Y**. Alternatively, click the **Redo** button  on the Main Toolbar. You can redo multiple times until there are no more redo items in the undo history.

Availability of undo and redo

You can undo and redo any changes made to the scene using the Select, Move, Rotate or Scale tools. You can also undo most changes made to properties using the Property Editor.

You **cannot** undo operations such creating or deleting objects, or cutting, pasting or deleting objects.

The undo history is **cleared** when you change the active tool. For example, if you use the Move tool to move objects in the scene, each move operation is added to the undo history. These operations can then be undone. However, if you then change the active

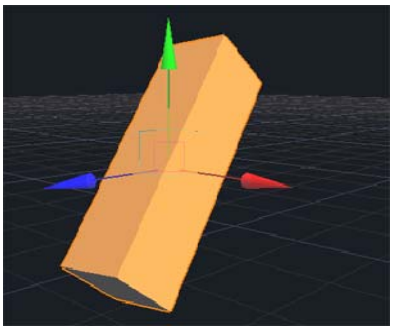
tool to the Rotate tool, the undo history is cleared. You will no longer be able to undo the earlier operations made using the Move tool.

To view the undo history:

- Select **Edit > Undo History…** to display the **Undo History** dialog.
- The **Undo stack** list shows the operations currently available in the undo history. The **Redo stack** list shows the operations currently available in the redo history. As operations are successively undone, they move from Undo stack to the Redo stack.
- You can clear the undo history by clicking **Clear history**.
- By default, the undo history has a limit of 1000 items. You can change this limit by editing the **Undo limit** value. If you do not want any limit on the undo history, turn off **Limit undo**.

Moving objects

One of the most common tasks when editing a scene is to **move** objects.



To activate the Move tool:

- To activate the Move tool, select **Edit > Move**. The keyboard shortcut is **Q**.

  Alternatively, you can click the **Move** button  in the Main Toolbar. When the Move tool is active, it will be displayed in the Main Toolbar using the **selection colour**.

To select objects using the Move tool:

- You can select, box select, additively select and toggle select in the viewports using the Move tool. Most of the time, selection using the Move tool works in the same manner as selection using the Select tool.

- However, in some cases, when an object is clicked, a **different** object is selected by the Move tool. For example, characters cannot be edited from within scenes. You can move characters themselves, but you cannot move the individual objects that make up a character. If you select a mass object or a collision object of a character, for example, the entire character becomes selected.
- When one or more objects are selected with the Move tool active, the **Move tool manipulator** is displayed in the viewports. It is displayed at the position of the first selected object. However, keep in mind that it affects **all** selected objects. You can use the Move tool manipulator in any of the viewports.

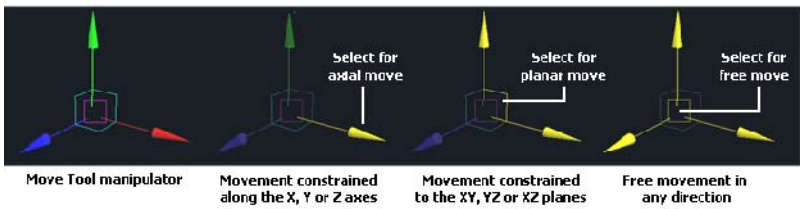
Multiple viewports

When using the Move tool, it is often useful to display multiple viewports. To display multiple viewports, click the **Select Multiple Viewports** button  in the Main Toolbar.

Using the Move tool

You can use the **Move** tool to freely move one or more objects in the virtual world.

You can also use the Move tool to move objects along a **specific direction** or in a **specific plane**. This is called a **constrained move**. You can perform constrained moves along any of the **global** X, Y or Z directions, or in any of the global XY, YZ or XZ planes. You can also perform constrained moves in the **local** X, Y or Z directions, or in the local XY, YZ or XZ planes of the first selected object.



To move objects in any direction:

- To move the selected objects freely in space, select inside the **purple square** manipulator and drag the mouse. The entire Move tool manipulator is displayed using a yellow highlight colour when you are freely moving selected objects.

To move objects in a specific direction:

- To move the selected objects along the **X direction**, select the **red** arrow manipulator and drag the mouse.
- To move the selected objects along the **Y direction**, select the **green** arrow manipulator and drag the mouse.
- To move the selected objects along the **Z direction**, select the **blue** arrow manipulator and drag the mouse.

When you are performing a constrained move along a particular direction, the corresponding arrow manipulator is displayed using a yellow highlight colour. Make sure correct arrow is highlighted before starting to drag.

To move objects in a specific plane:

- To move the selected objects in the **XY plane**, select the segment of the outer **light blue (aqua)** box manipulator between the **red** and **green** arrow manipulators and drag the mouse.
- To move the selected objects in the **YZ plane**, select the segment of the outer **light blue (aqua)** box manipulator between the **green** and **blue** arrow manipulators and drag the mouse.
- To move the selected objects in the **XZ plane**, select the segment of the outer **light blue (aqua)** box manipulator between the **red** and **green** arrow manipulators and drag the mouse.

When you are performing a constrained move in the particular plane, the two corresponding arrow manipulators that define the plane are displayed using a yellow highlight colour. Make sure correct arrows are highlighted before starting to drag.

To change the coordinate system of a constrained move:

- By default, the Move tool is aligned along the global coordinate system.
- To toggle the Move tool coordinate system so that it is aligned with the local coordinate system of the first selected object, select **Edit > Toggle Coordinate System**. The keyboard shortcut is **X**. The Move tool manipulator updates to reflect the active coordinate system.

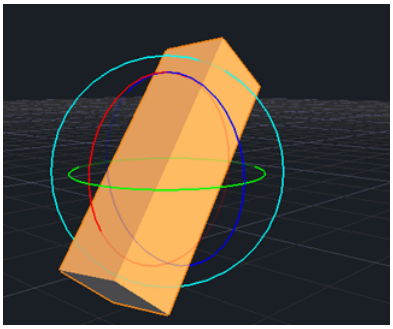
- When you change the coordinate system for the Move tool, this setting becomes the default for subsequent uses of the Move tool. It also becomes the default for the **Rotate** tool.

To cancel a move:

- A move can be cancelled at any time while you are dragging the Move tool manipulator.
- To cancel a move, **right-click**. This moves the Move tool manipulator back to its original position at the start of the move.

Rotating objects

A common task when editing a scene is to **rotate** objects.



To activate the Rotate tool:

- To activate the Rotate tool, select **Edit > Rotate**. The keyboard shortcut is **W**.

  Alternatively, you can click the **Rotate** button  in the Main Toolbar. When the Rotate tool is active, it will be displayed in the Main Toolbar using the **selection colour**.

To select objects using the Rotate tool:

- You can select, box select, additively select and toggle select in the viewports using the Rotate tool. Most of the time, selection using the Rotate tool works in the same manner as selection using the Select tool.
- However, in some cases, when an object is clicked, a **different** object is selected by the Rotate tool. For example, characters cannot be edited from within scenes. You can rotate characters themselves, but you cannot rotate the individual objects that make up a character. If you select a mass object or a collision object of a character, for example, the entire character becomes selected.

- When one or more objects are selected with the Rotate tool active, the **Rotate tool manipulator** is displayed in the viewports. It is displayed at the position of the first selected object. However, keep in mind that it affects **all** selected objects. You can use the Rotate tool manipulator in any of the viewports.

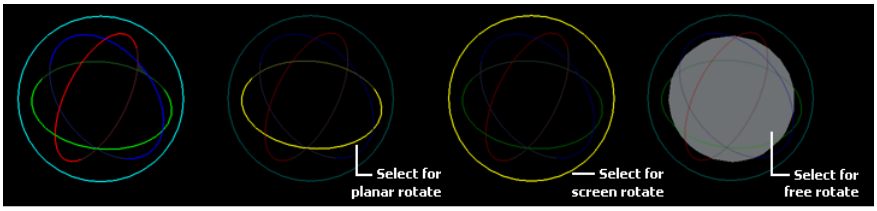
Multiple viewports

When using the Rotate tool, it is often useful to display multiple viewports. To display multiple viewports, click the **Select Multiple Viewports** button  in the Main Toolbar.

Using the Rotate tool

You can use the **Rotate** tool to freely rotate one or more objects in the virtual world.

You can also use the Rotate tool to rotate objects in about a **specific axis or plane**. This is called a **constrained rotate**. You can perform constrained rotations around any of the **global** X, Y or Z directions. You can also perform constrained rotations around any of the **local** X, Y or Z directions of the first selected object. Additionally, you can perform a constrained rotation in the **viewport plane**.



To rotate objects in any direction:

1. Rotating objects freely in any direction of often called **trackball rotation**. Select somewhere in the region **near the centre** of the manipulator. Be careful to avoid any of the specific manipulator circle controls. If you select too far from the centre of the manipulator, you will enter box selection mode.
2. Drag the mouse to freely rotate the selected entities. During trackball rotation, a grey filled circle is displayed in the interior of the Rotate tool manipulator.

To rotate objects around a specific axis or plane:

- To rotate the selected objects around the **X axis**, select the **red** circle manipulator and drag the mouse.

- To rotate the selected objects around the **Y axis**, select the **green** circle manipulator and drag the mouse.
- To rotate the selected objects around the **Z axis**, select the **blue** circle manipulator and drag the mouse.
- To rotate the selected objects in the **viewport plane**, select the outer **light blue (aqua)** circle manipulator and drag the mouse.

When you are performing a constrained drag around a particular axis or in a particular plane, the corresponding circle manipulator is displayed using a yellow highlight colour. Make sure correct circle manipulator is highlighted before starting to drag.

To change the coordinate system of a constrained rotate:

- By default, the Rotate tool is aligned along the global coordinate system.
- To toggle the Rotate tool coordinate system so that it is aligned with the local coordinate system of the first selected object, select **Edit > Toggle Coordinate System**. The keyboard shortcut is **X**. The Rotate tool manipulator updates to reflect the active coordinate system.
- When you change the coordinate system for the Rotate tool, this setting becomes the default for subsequent uses of the Rotate tool. It also becomes the default for the **Move** tool.

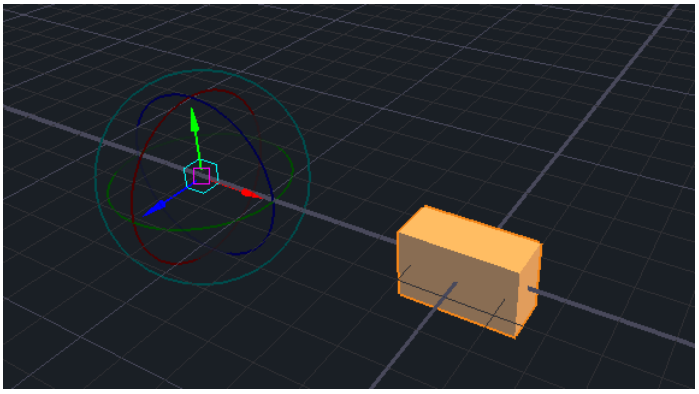
To cancel a rotation:

- A rotation can be cancelled at any time while you are dragging the Rotate tool manipulator.
- To cancel a rotation, **right-click**. This moves the Rotate tool manipulator back to its original position at the start of the rotation.

Changing the rotation pivot point

When you use the **Rotate** tool to rotate objects, they rotate about the **rotation pivot point**. For most objects, the default rotation pivot point corresponds to the centre of the object. For joints, the default pivot point corresponds to the joint connector position. If you are rotating multiple objects, the rotation pivot point used is the default pivot point of the first selected object.

When you are using the Rotate tool, you can change the location of the pivot point. This new pivot point is only used while the Rotate tool is active. If you change the set of selected entities, the pivot point is **reset** back to the default. If you make changes to the pivot point position, these are transitory and **not** permanently stored with the associated object.

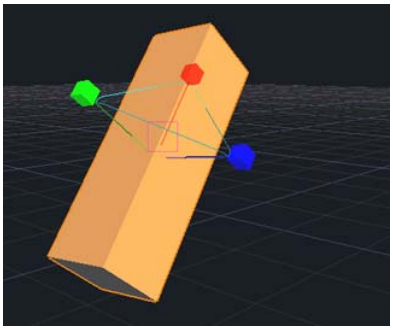


To change the rotation pivot point position:

1. Ensure the **Rotate** tool is active.
2. Hold down **Ctrl** and **right-click** anywhere in the viewport. You do not need to click directly over the Rotate tool manipulator. This temporarily places the Rotate tool into the **Adjust Pivot Point mode**.
3. In this mode, the Rotate manipulator is temporarily disabled, and a **Move** tool manipulator is displayed. You can use the Move tool manipulator to change the location of the rotation pivot point. This Move tool manipulator works exactly as for the standard Move tool, except that right-click cancellation does not operate.
4. To exit Adjust Pivot Point mode, **right-click** anywhere in the viewport. This reactivates the Rotate tool manipulator, and moves it so that it corresponds to the new pivot point position.
5. Any subsequent rotations using the Rotate tool will rotate the selected objects about the new pivot point.
6. The pivot point position will be reset if you make any changes to the set of selected entities. It is also reset if you toggle the coordinate system from local coordinates to global coordinates or vice versa.

Scaling objects

A common task when editing a scene is to **scale** objects. This changes the size of the object. You can change the overall size of the object, or just the size in one direction at a time.



To activate the Scale tool:

- To activate the Scale tool, select **Edit > Scale**. The keyboard shortcut is **E**.

  Alternatively, you can click the **Scale** button  in the Main Toolbar. When the Scale tool is active, it will be displayed in the Main Toolbar using the **selection colour**.

To select objects using the Scale tool:

- You can select, box select, additively select and toggle select in the viewports using the Scale tool. Most of the time, selection using the Scale tool works in the same manner as selection using the Select tool.
- Different object types have different rules governing scaling. Characters, joints, graphical objects and spherical mass and collision objects can only be scaled **uniformly**. Cylindrical and sphyl mass and collision objects can only be scaled in the **axial** and **radial** directions. Box mass and collision objects can be scaled in all directions **independently**.
- When you are editing a **scene**, only child mass objects, collision objects and graphical objects of the Environment character may be scaled.
- When you are in **Character Edit Mode**, you can scale child mass objects, collision objects, graphical objects and joints of the active character. In addition, if you are editing a simulation-reference character pair, you can also scale the **reference** character.
- When one or more objects are selected with the Scale tool active, the **Scale tool manipulator** is displayed in the viewports. It is displayed at the position of the

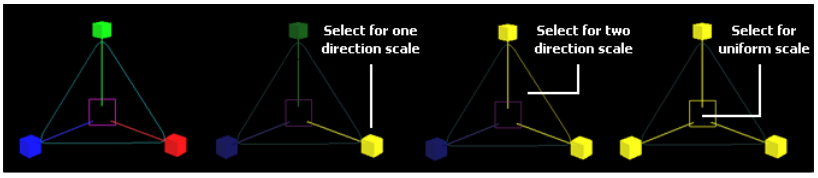
first selected object. However, keep in mind that it affects **all** selected objects. You can use the Scale tool manipulator in any of the viewports.

Multiple viewports

In order to scale objects using the viewports, you must first activate the **Scale** tool. When using the Scale tool, it is often useful to display multiple viewports. To display multiple viewports, click the **Select Multiple Viewports** button  in the Main Toolbar.

Using the Scale tool

You can use the **Scale** tool to change the size of one or more objects. The Scale tool always uses the **local** coordinate system of any selected objects. You cannot change the coordinate system used by the scale tool.



Limitations on scaling objects

You can use the Scale tool to scale most objects **uniformly** in all directions. You can also scale some object types along a **specific direction** or in a **specific plane**. This is called a **constrained scale**.

Different object types have their own rules governing how they can be scaled:

- **Spherical** mass and collision objects can only be scaled uniformly.
- **Cylindrical** mass and collision objects can only be scaled axially or radially.
- **Sphyl** mass and collision objects can only be scaled axially or radially.
- **Box** mass and collision objects can be scaled in any direction.
- **Characters** can only be scaled uniformly.
- **Joints** can only be scaled uniformly.
- **Graphical objects** can only be scaled uniformly.

Also, keep in mind that characters and their child objects—such as joints, mass objects, collision objects and graphical objects—can only be scaled in Character Edit Mode. When

you are editing a scene, only objects belonging to the Environment character can be scaled.

To scale objects in all directions:

- To scale the selected objects in all directions, select inside the **purple square** manipulator and drag the mouse. The entire Scale tool manipulator is displayed using a yellow highlight colour when you are scaling in all directions. The selected objects will be scaled equally in their local X, Y and Z coordinates.

To scale objects in a specific direction:

- To scale the selected objects along their local **X direction**, select the **red** cylinder manipulator and drag the mouse.
- To scale the selected objects along their local **Y direction**, select the **green** cylinder manipulator and drag the mouse.
- To scale the selected objects along their local **Z direction**, select the **blue** cylinder manipulator and drag the mouse.

When you are performing a constrained scale along a particular direction, the corresponding cylinder manipulators are displayed using a yellow highlight colour. Make sure correct manipulator highlighted before starting to drag.

Keep in mind that some objects—such as characters, joints, graphical objects and spherical mass and collision objects—are always scaled **uniformly**, even when you are using the Scale tool to scale in a specific direction.

To scale objects in a specific plane:

- To scale the selected objects in the **XY plane**, select the segment of the outer **light blue (aqua)** box manipulator between the **red** and **green** arrow manipulators and drag the mouse.
- To scale the selected objects in the **YZ plane**, select the segment of the outer **light blue (aqua)** box manipulator between the **green** and **blue** arrow manipulators and drag the mouse.
- To scale the selected objects in the **XZ plane**, select the segment of the outer **light blue (aqua)** box manipulator between the **red** and **green** arrow manipulators and drag the mouse.

When you are performing a constrained scale in a particular plane, the two corresponding arrow manipulators that define the plane are displayed using a yellow highlight colour. Make sure correct arrows are highlighted before starting to drag.

Keep in mind that some objects—such as characters, joints, graphical objects and spherical mass and collision objects—are always scaled **uniformly**, even when you are using the Scale tool to scale in a specific plane.

To cancel a scale:

- A scale can be cancelled at any time while you are dragging the Scale tool manipulator.
- To cancel a scale, **right-click**. This moves the Scale tool manipulator back to its original position at the start of the scale.

Application keyboard shortcuts

Using keyboard shortcuts for application-level operations is an efficient way to work with *endorphin*.

Summary of keyboard shortcuts

- **F1** to display the online help.
- **N** to show and hide the Node View.
- **Alt+F** to toggle full-screen mode.
- **Alt+F4** to close *endorphin*.
- **Ctrl+Shift+L** to lock and unlock the layout.

File keyboard shortcuts

Using keyboard shortcuts for file operations is an efficient way to work with *endorphin*.

Summary of keyboard shortcuts

- **Ctrl+N** to create a new scene.
- **Ctrl+O** to open an existing scene.
- **Ctrl+S** to save the scene.

Edit keyboard shortcuts

Using keyboard shortcuts for editing scenes is an efficient way to work with *endorphin*.

Summary of keyboard shortcuts

- **R** to activate the Select tool.
- **Q** to activate the Move tool.
- **W** to activate the Rotate tool.
- **E** to activate the Scale tool.
- **A** to activate the Pose Move tool.
- **S** to activate the Pose Rotate tool.
- **X** to toggle between global coordinate system and local coordinate system.
- **Ctrl+X** to cut the selected objects.
- **Ctrl+C** to copy the selected objects.
- **Ctrl+V** to paste.
- **Delete** to delete.
- **Ctrl+Z** to undo.
- **Ctrl+Y** to redo.

Summary of mouse shortcuts

- **Double-click** on a Move manipulator to activate the Rotate tool.
- **Double-click** on a Rotate manipulator to activate the Scale tool.
- **Double-click** on a Scale manipulator to activate the Move tool.
- **Double-click** on a Pose Move manipulator to activate the Pose Rotate tool.
- **Double-click** on a Pose Rotate manipulator to activate the Pose Move tool.

Custom viewport cameras

endorphin scenes are displayed in viewports using one of the **viewport cameras**.The standard viewport cameras are the Persective View, Left View, Right View, Top View, Bottom View, Front View and Back View cameras. These are available in every viewport.

You can also create additional **custom viewport cameras**. Custom cameras are useful when you want to view the scene from a particular angle, or when you want to use a particular lens setting that matches your external requirements.

To create a custom viewport camera:

- Select **Environment > Create Camera**. A new custom camera is created at the origin. The camera is added to **the Node View**, and is also added to the viewport context menu (under the **Viewport** menu item).
- You can create up to **ten** custom cameras.

To select a custom viewport camera:

- The easiest way to select a custom camera is via the Node View. Custom cameras are listed after the characters.
- You can also select cameras using their corresponding **camera graphic** in the viewport. If you have turned off the Camera Display setting in the Property Editor, the camera graphic is not displayed in the viewport, and you will not be able to select it.

To view the scene using custom viewport camera:

- Right-click in the viewport, and select **Viewport**, then choose the custom camera from the list of cameras. The custom cameras are listed below the standard cameras. The viewport label will update to reflect the name of the active camera.

To adjust the position and orientation of a custom viewport camera:

- If you are **not** currently viewing the scene through the custom viewport camera, select the camera using the Node View or its camera viewport graphic. You can then use the **Move** and **Rotate** tools to reposition the camera.
- If you **are** currently viewing the scene through the custom viewport camera, any viewport **pan**, **rotate** or **zoom** operations that you perform will affect the position and orientation of the active viewport camera.

To modify the properties of a custom camera:

- Select the camera using the Node View or the viewport. You can then modify many of the camera settings, such as its focal length, front and back cutting planes, picture ratio and film gate display setting using the Property Editor.

To delete a custom viewport camera:

- Select the camera using the Node View or the viewport, and select **Edit > Delete** or **Edit > Cut**. Alternatively, press the **Delete** key or use **Ctrl+X**.

What are animated cameras?

A common use of *endorphin* is to generate animation that will be used to animate characters and objects in other animation systems such as Maya and 3dsMax. In many cases, the resulting animation will be rendered in these systems using **animated custom cameras** that have been keyframed to move along a specific path during the course of the scene.

endorphin ships with various scripts to convert animated camera motion from these systems into *endorphin* **camera files**. These are plain text files with the extension **.nmcam**.

- The Maya MEL script **endorphin_Exporter.mel** can be run in Maya to save Maya animated cameras as *endorphin* **.nmcam** camera files. This file is shipped in the **Resources\Third Party\Scripts\Maya** folder.
- The 3dsMax MAXscript **endorphin_CameraExporter.ms** can be run in 3dsMax to save 3dsMax animated cameras as *endorphin* **.nmcam** camera files. This file is shipped in the **Resources\Third Party\Scripts\3dsMax** folder.

To import a camera:

1. Select **File > Import…** to display the **Select File To Import** dialog. The keyboard shortcut is **I**.
2. Select the *endorphin* **.nmcam** camera file that you want to import.
3. Click **Open** to display the **Import Options** dialog.
4. Click **OK** to import the camera. A new **custom viewport camera** is created in the scene.

To view the scene with a custom viewport camera:

- When a camera has been imported into a scene, you can use it to view the scene as an alternative to the usual Top, Left, Front, Back, Top, Bottom and Perspective Views.

- To view the scene with an imported camera, **right-click** and select the camera by name. By default, cameras are imported with the names Camera01, Camera02 and so on. The list of imported cameras is at the bottom of the context menu. When you are viewing a scene with an imported camera, the camera name appears in the **viewport label**.
- When you are viewing a scene with an imported camera, grey **bands** may be displayed at the top and bottom, or left and right, of the viewport. You will see these bands if the **picture ratio** of the camera is different to the current aspect ratio of the viewport. The bands indicate regions that will not be rendered in the final animation system.
- When you are viewing a scene with an imported camera, the imported camera keyframes define the position and orientation of the camera for all frames. When you simulate a scene, play back a simulation, or manually scrub through the timeline, the camera position and orientation automatically update for each frame. Keep in mind that you **cannot** pan, rotate or zoom the viewport. This is because the camera position and orientation are completely defined by the imported camera keyframes.

To select a camera:

- Cameras are displayed in viewports using the **camera** graphic. You can select cameras by **clicking** on them in the viewport. You can also **box-select** and **Ctrl-select** in the viewport.
- Cameras are displayed in the **Node View** using the camera icon. You can select cameras by **clicking** on them in the Node View. Note that you should not attempt to hide a camera by right-clicking on a camera node in the Node View and selecting **Hide All**. The corresponding command **Show All** does not currently apply to cameras, so you will not be able to show the camera after it has been hidden.
- If a camera is currently active in a viewport, **right-click** in the viewport and select **Camera** to select the corresponding camera.

To edit a camera:

- Once you have selected a camera, its properties are displayed in the Property Editor. You can edit some of these properties to modify the camera settings.

  Do not use the Move or Rotate tools to move or rotate the camera. Any changes you make to the position or orientation of the camera are ignored. The position and orientation of a camera are entirely defined by the data in their source **.nmcam** file.

To delete a camera:

- Select the camera that you want to delete, and then select **Edit > Delete**. Alternatively, press **Delete**.

Introducing the video plane

The **video plane** is an animated image that is displayed in the background of an *endorphin* viewport. Video planes are always used in conjunction with imported animated cameras. By projecting existing live-action or animated footage onto the backplane, you are able to visualise an interactive composite of *endorphin* characters simulated in their final setting.

To use the *endorphin* video plane, you must have access to a **.avi file** that contains video data that matches the motion of the imported camera. That is, the .avi file must have been rendered or filmed using the same camera settings as were used to create the corresponding **.nmcam** camera file.

To use the video plane:

1. If you are using Maya, export camera motion from Maya using the **endorphin_Exporter.mel** MEL script. This creates a **.nmcam** *endorphin* camera file. Alongside the MEL script file is an accompanying text file, **endorphin_Exporter_ReadMe.txt**, that includes more detailed help.

   If you are using 3dsMax, export camera motion from 3dsMax using the **endorphin_CameraExporter.ms** MAXscript. This creates a **.nmcam** *endorphin* camera file. Alongside the script file is an accompanying text file, **endorphin_MaxScripts.txt**, that includes more detailed help.

2. Render the Maya scene or 3dsMax scene from the point-of-view of the same camera that was exported in the previous step. Save this render as a **.avi** video file.

3. Import the generated **.nmcam** camera file into *endorphin*.

4. Activate this camera by **right-clicking** in the viewport and selecting the camera name from the context menu.

5. Select this camera by **right-clicking** in the viewport and selecting **Select Camera**. This displays the camera properties in the Property Editor.

6. Turn on the **Video Plane** setting in the Property Editor. This will display the video plane properties in the Property Editor. It will also display a black-and-white chequered texture in the background of the viewport.

7. Click on the **[…]** button in the **File** property to browse for a **.avi** file to use as the video backplane file. When you have located a valid .avi file, endorphin will display the first frame of this .avi file in the viewport background.

   When you simulate, replay a simulation or manually scrub through the timeline, *endorphin* will display the corresponding frame in the video backplane .avi file that matches the current *endorphin* frame.

8. Use the **Start Frame** and **End Frame** properties to specify when to start and stop displaying the .avi file.

   Use the **Frame Rate** property to synchronise the frame rate of the .avi file with the frame rate of the simulation.

   Use the **Frame Offset** property to specify the start frame to use in the .avi file.

   Use the **Offset X** and **Offset Y** properties to adjust horizontal and vertical position of the correctly position the .avi file relative to the viewport background.

   Use the **Scale X** and **Scale Y** properties to adjust the horizontal and vertical sizes of the .avi file.

Camera properties

Custom viewport cameras have the following properties.

Name

Range: Any unique name. **Default value:** Camera01.

Specifies the name of the camera. The camera name is used in the viewport label when that camera is active. In addition, the camera name appears in the viewport context menu.

Type

Range: Perspective. **Default value:** Perspective.

Specifies the camera type. Currently, the only setting is Perspective.

Mode

Range: Free. **Default value:** Free.

Specifies the camera mode. Currently, the only setting is Free.

Focal length

Range: Positive value. **Default value:** Imported.

Specifies the focal length of the camera, which effectively specifies the camera zoom factor.

Centre of interest

Range: Positive value. **Default value:** Imported.

Specifies the pivot position about which the camera pitches and yaws.

Front cut plane

Range: Positive value. **Default value:** Imported.

Specifies the front clipping plane. Objects that are closer to the camera than this distance will not be displayed when viewing the scene with this camera.

Back cut plane

Range: Positive value. **Default value:** Imported.

Specifies the back clipping plane. Objects that are further from the camera than this distance will not be displayed when viewing the scene with this camera.

Aperture X

Range: Positive value. **Default value:** Imported.

Specifies the horizontal filmgate aperture, which effectively specifies the width of the image at the film plane or render plane.

Aperture Y

Range: Positive value. **Default value:** Imported.

Specifies the vertical filmgate aperture, which effectively specifies the height of the image at the film plane or render plane. Cannot be edited in the scene. Can only be specified by the original animated camera.

Picture ratio

Range: Positive value. **Default value:** Imported.

Specifies the film aspect ratio, which is the ratio between the width and height of the image at the film plane or render plane.

Squeeze ratio

Range: Positive value. **Default value:** Imported.

Specifies the lens squeeze ratio, which is used for creating or correcting anemographic effects.

Camera visible

Range: On/Off. **Default value:** On.

Specifies whether the camera graphic is visible in the scene.

Frustrum visible

Range: On/Off. **Default value:** Off.

Specifies whether the view frustrum graphic is visible in the scene. The view frustrum is pyramid-shaped volume that emanates from the camera and defines which objects in the scene are visible to the camera.

Film gate display

Range: None/Mask/Line/Letterbox. **Default value:** Mask.

Specifies how the filmgate is displayed in *endorphin* viewports. The **Line** setting displays horizontal or vertical lines. The **Mask** setting displays horizontal or vertical lines, as well as semi-transparent mask regions. The **Letterbox** setting displays completely opaque horizontal or vertical mask regions.

Lock settings

Range: On/Off. **Default value:** On.

Specifies whether the other properties of the camera can be edited in the Property Editor. Used to avoid inadvertent changes to camera settings.

Video plane

Range: On/Off. **Default value:** Off.

Specifies whether the video plane is displayed in the viewport. Also used to specify whether to display the video plane properties of the camera in the Property Editor.

Position and orientation

Range: Unlimited values. **Default value:** 0.0.

Specifies the position and orientation of the camera in **World** coordinates. Note that the Reference Frame property is ignored. You cannot edit these settings to change the position or orientation of a camera. Similarly, you cannot use the Move or Rotate tools to change the position or orientation of a camera. The position and orientation of a camera are entirely defined by the keyframe data that is contained in the corresponding **.nmcam** camera file.

Chapter 6

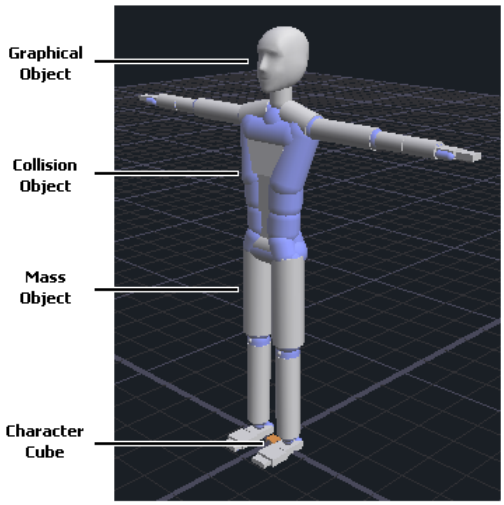
Characters

What is an *endorphin* character?

The *endorphin* virtual world is populated by **characters**. You can use characters to represent human or humanoid participants in the virtual world. More generally, you can use characters to represent other scene elements such as animals and vehicles. In addition, the environment itself is also modeled as an **environment character**.

Characters are combinations of mass objects and collisions that have been arranged using a hierarchy of **joints**. Each joint connects two mass objects together. You can use different types of joints, such as **skeletal joints** for general motion between the mass objects, or **hinge joints**, to constrain motion between the mass objects along a single axis. Each joint has a corresponding **joint limit**, that specifies the range of motion. Collections of mass objects and collision objects connected by joints are sometimes called **articulated structures**.

Each character also has a **character cube** rendered into the viewports. You can select, move and rotate characters by selecting the character cube, as an alternative to selecting the character itself.



Character cubes

Characters have a **local coordinate system**. You can edit the location and orientation of the local coordinate system using Character Edit Mode. For the standard simulation character, the origin of the local coordinate system is located between the feet of the character in its T-pose.

The origin of the local character coordinate system is diaplayed in the viewport using a small cubic helper graphic. This graphic is called the **character cube**. Each character in the scene—except for the Environment character—has a character cube. Character cubes are assigned a unique colour.

Using character cubes

- You can **select** characters by selecting their character cubes in the viewport.
- If you **rotate** a character using the Rotate tool, the character will rotate about its local character origin, as defined by the location of its character cube.
- You can toggle the visibility of character cubes by toggling **View > Character Cubes**, or by right-clicking in the viewport and toggling **Display > Character Cubes**.
- When you are creating **custom characters**, you can edit the position and orientation of their character cube—which defines the relative position and orientation of their local coordinate system—using **Character Edit Mode**.
- When you are **exporting** animation data, you can choose whether to export the **root joint** translation and rotation channels in terms of the world coordinate system, or in terms of the local character coordinate system.

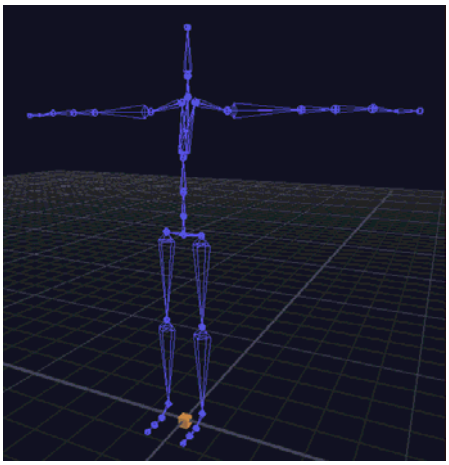
Standard simulation character

When you create a new scene in *endorphin*, it is initially populated by an instance of the **standard simulation character**. This character represents an average-sized adult male human. Like all characters, the standard simulation character is a joint hierarchy, with a mass objects associated with each joint, and one or more collision objects associated with each mass object.

Standard character joint hierarchy

The standard character has a skeleton root joint called **Root**.

From the Root, the joint hierarchies are as follows. All joints are skeletal joints, except for the LeftFingersJoint and RightFingersJoint, which are hinge joints. Any joints with an **–EndJoint** suffix are simple joints. Keep in mind that the joints along the right arm and right leg have the same sequence as the joints along the left arm and left leg, except that **Left-** prefixes are replaced with **Right-**.



Joints along the spine:

Root > LowerSpineJoint > MiddleSpineJoint > UpperSpineJoint > LowerNeckJoint > UpperNeckEndJoint

Joints along the left arm:

Root > LowerSpineJoint > MiddleSpineJoint > UpperSpineJoint > LeftClavicleJoint > LeftShoulderJoint > LeftElbowJoint > LeftForearmTwistJoint > LeftWristJoint > LeftFingersJoint > LeftFingersEndJoint

Joints along the left leg:

Root > LeftHipJoint > LeftKneeJoint > LeftAnkleJoint > LeftMidFootJoint > LeftToesJoint > LeftToesEndJoint

Note The standard character has an additional forearm twist joint, and also a multiple-joint foot articulation. These additional joints play an important role in generating high-quality animation. For example, the foot articulation helps ensure minimal foot slip during simulations, which increases their corresponding realism.

Joint limits and joint limit indicators

Skeletal joints and hinge joints have a corresponding joint limit with the same name as the joint, except that the **–Joint** prefix is replaced with **–JointLimit**. Joint limits appear as a child node of the corresponding joint in the Node View.

Skeletal joints and hinge joints also have a corresponding joint limit indicator with the same name as the joint, except that the **–Joint** suffix is replaced with **–JointLimitOffset**. Joint limit indicators appear as a child node of the corresponding joint in the Node View.

Standard character mass and collision objects

Skeletal joints, hinge joints and the skeleton root joint all have a single corresponding mass object. Mass objects all have one or more corresponding collision objects. Mass object have the suffix **–Mass**. Collision objects have the suffix **–Collision.**

Along the standard character limbs, there is generally a single collision object per mass object. In other places, such as the torso, there are as many as eight child collision objects. This is to fill out the torso muscle system in a realistic manner

Working with mass and collision objects

The standard character is designed so that the mass objects and collision objects associated with each joint reflect the mass and volume associated with the bone or bones that are associated with the joint. The construction of the standard character means that bones, and therefore the corresponding mass and collision objects, are always **downstream** of their corresponding joint.

For example, the **left hip joint** has mass and collision objects that define the **upper leg (thigh)** mass and volume. Similarly, the **left knee joint** has mass and collision objects that define the **lower leg (calf)** mass and volume.

Standard and custom characters

You can populate the virtual world with any number of **characters**. Characters may represent humans or humanoids, or may represent animals, vehicles or other types of inanimate objects.

By default, new *endorphin* scenes contain an **environment** character. This character cannot be deleted. New scenes also contain a single instance of the **standard simulation character**, arranged at the centre of the virtual world in a T-pose. You can delete this character. You can also add and delete other characters.

Characters are stored in **.nmc** character binary files. When you add a character to the scene, the character definition is **copied** into the scene.

Standard simulation character

The standard simulation character is stored in the file **Resources\Characters\Standard\Standard Simulation Character.nmc**.

Custom simulation characters

endorphin ships with some example simulation characters that have been **derived** from the standard simulation character. These are often called **custom simulation characters**. These are installed into the folder **Resources\Characters\Custom**, and include a larger character and a female character. You can also create your own simulation characters and store them in different locations.

Adding characters to scenes

By default, new *endorphin* scenes contain an **environment** character. This character cannot be deleted. New scenes also contain a single instance of the **standard simulation character**, arranged at the centre of the virtual world in a T-pose. You can delete this character. You can also add and delete other characters.

To add a character:

1. Select **Character > Add Character…**. This displays the **Add Character** dialog.

   Alternatively, click the **Add Character** button  in the Main Toolbar.

2. The list of characters will **always** include the **Standard Simulation Character**. This is the standard *endorphin* character, and represents an average-sized adult human male.

   The list may also contain a list of **Simulation Characters**. These are characters that have been derived from the **standard simulation character**. Simulation characters have the same joint hierarchy as the standard simulation character, but can have a different shape and size.

   The list may also contain a list of **Prop Characters**. These are characters that have been derived from the **standard prop character**. Prop characters are used to represent objects such as animals, vehicles and weapons.

3. The **Path** text box contains the location of your custom character folder. Click the **[…]** button to display the **Browse For Folder** dialog. You can browse to a different character folder. The list of Simulation Characters and Prop Characters updates to reflect the characters that the system finds in the character folder that you have specified.

The Path character folder is remembered by *endorphin*. This means that once you have browsed to a character folder once, you do not have to keep on browsing to the same location. Regardless of the character folder you have selected, the standard simulation character is always available in the character list.

4. Click **OK** to add the selected character to the scene. Alternatively, double-click on a character to add it to the scene.

Intersecting characters

New characters are always created at the **origin** of the virtual world. This can sometimes be confusing, as it means that you can inadvertently add multiple characters to a scene at the same position. It is good practice to always **move** new characters away from the origin of the virtual world **before** adding further characters.

If you inadvertently attempt to simulate with multiple intersecting characters in your scene, the simulation rate will fall rapidly as the characters involved violently decouple. If this occurs, stop the simulation and manually separate the intersecting characters. Likewise, it is important to ensure that no characters are intersecting with any of the **Environment** character collision objects before running a simulation.

Learning Edition characters

Keep in mind that characters created in *endorphin* Learning Edition cannot be added to scenes in *endorphin* or *endorphin* Educational Edition, and vice versa.

Deleting characters from scenes

By default, new *endorphin* scenes contain an **environment** character. This character cannot be deleted. New scenes also contain a single instance of the **standard simulation character**, arranged at the centre of the virtual world in a T-pose. You can delete this character. You can also add and delete other characters.

To delete a character:

1. Use the Timeline Editor or the Node View to select the character or characters that you want to delete. Alternatively, use the viewports to select the desired characters.
2. Select **Edit > Delete** to delete the selected characters. Alternatively, press the **Delete** key.

   Note You cannot delete the **Environment** character.

Naming characters

When you add a character to a scene, it is given a unique name based upon the corresponding **.nmc** filename. For example, if you add two instances of **Soldier.nmc** to a scene, their default names might be **Soldier01** and **Soldier02**. If you add instances of the standard simulation character to a scene, they will be given generic names such as **Character01** and **Character02**.

You can edit these default character names. This is a useful way of customising a scene, as it allows you to give characters more meaningful names that reflect their role in a scene. It is particularly useful if you have a scene with a large number of prop, simulation and reference characters. User-defined character names appear in the Timeline Editor, Node View and Property Editor.

Character names can have any combination of alphanumeric characters or spaces. The only limitations are that all characters mus have unique names, and names cannot be empty. Note that you cannot rename the Environment character.

Internal and external names

When you edit the name of a character, you are actually editing its **external** name. Characters also have a simple, sequential **internal** character name that cannot be edited.

If you hover over a character name in the **Timeline Editor**, the tooltip will also display its corresponding internal and external character names. For example, if the tooltip displays **Character02: 'Helicopter'**, this indicates that the character's internal name is **Character02**, and its external name is **Helicopter**.

Fully-qualified names

All entities in *endorphin* have a **fully-qualified** name that defines their specific entity name, as well as their entity type and their character. For example, the left elbow joint of Character02 has the fully-qualified name **Character02.Joints.LeftElbowJoint**. Fully-qualified entity names always used the **internal** character name, not its external, user-defined, name.

Fully-qualified entity names are used in collection properties in the **Property Editor**. For example, the **Target Joints** property of an active pose event is a collection property that contains a list of fully-qualified joint names.

Selecting characters

You can select characters in a number of ways:

Using the viewport:

- You can select any character (except the Environment character) in a **viewport** by clicking its **character cube** graphic. Hold down the **Ctrl** key to select multiple characters.
- You can also select any character by right-clicking in a viewport and selecting **Select Character**, and then selecting one of the characters from the character submenu.

Using the Node View:

- You can select any character using the **Node View** by clicking or right-clicking on the corresponding character node.
- Hold down the **Ctrl** key to select multiple characters. Note that if you multi-select characters in the Node View, only the **last** selected character is highlighted in the Node View using the selection colour. Nevertheless, multiple characters are still selected.

Using the Timeline Editor:

- You can select any character using the **Timeline Editor** by clicking or right-clicking on the corresponding character name.
- Hold down the **Ctrl** key to select multiple characters. Note that if you multi-select characters in the Timeline Editor, only the **first** selected character is highlighted in the Timeline Editor using the selection colour. Nevertheless, multiple characters are still selected.

Showing and hiding characters

You can show and hide characters in a number of ways.

Using the viewport:

- You can show or hide any character by right-clicking in a **viewport** and selecting **Show/Hide Character**, and then selecting one of the characters from the character submenu. The entire character (including all its mass objects, collision objects, graphical objects and joints) is shown or hidden. Joint limits are must be shown individually.

Using the Node View:

- You can show or hide any character using the **Node View** by right-clicking on the corresponding character name, and selecting **Show All** or **Hide All**. The entire character (including all its mass objects, collision objects, graphical objects, and joints) is shown or hidden. Joint limits must be shown individually.
- Keep in mind that right-clicking on a character name in the Node View and selecting **Show** or **Hide** will only show or hide the **character cube** of the character. To show or hide the entire character you must use the **Show All** and **Hide All** commands.

Using the Timeline Editor:

- You can show or hide any character using the **Timeline Editor** by right-clicking on the corresponding character name, and selecting **Show Character** or **Hide Character**. The entire character (including all its mass objects, collision objects, graphical objects and joints) is shown or hidden. Joint limits must be shown individually.

Introducing character posing

When you add a character to a scene, it is initially in its **default pose**. For the **standard simulation character**, the default pose is a **T-pose**. When you create custom characters, you can save them with a different default pose.

When you use the Move or Rotate tools with a character, the entire character is moved and rotated. These tools do not change the character pose itself.

Making changes to the character pose is called **posing** the character. To make changes to character poses, you need to use the **Pose Move** and **Pose Rotate** tools.

Poses and keyframes

You can always pose any character at the **start frame**. If you have already simulated (so that there are already some frames in the frame buffer), then you can create poses for any character at other frames, by moving the time slider to the desired frame, and then posing characters as required.

When you pose a character, a blue **keyframe marker** is added to the Timeline Editor for that character and frame. This means that endorphin has stored the pose as a keyframe. If you pose a character at a frame for which it already has a keyframe, the keyframe is replaced. Once you have created a pose keyframe at some particular frame, clearing the frame buffer does not affect the keyframe.

For more information on how endorphin uses pose keyframes, see the **Simulation** topics in this guide.

Dynamic posing

To edit character poses, use the **Pose Move** and **Pose Rotate** tools. These pose tools use a process called **dynamic posing**. You can use dynamic posing to pose any character in the scene, including the **environment** character.

Preventing object penetration

One advantage of using dynamic posing is that it ensures that collision objects do not **interpenetrate**. When you use the Move, Rotate and Scale tools, no checks are made to ensure that collision objects in the scene are not penetrating. If you simulate a scene in which collision objects are penetrating at the start frame, they will tend to accelerate rapidly apart, which can destabilise the simulation.

When you use the Pose Move and Pose Rotate tools, checks are made to ensure that collision objects do not penetrate. For example, you could use the Pose Move tool to rapidly stack a set of box collision objects to form a wall of objects.

Freezing and locking

Dynamic posing allows you to work in a manner which is a hybrid of traditional forward kinematics (FK) and inverse kinematics (IK) approaches to posing, particularly when you use **freezing** to control which objects are affected by the pose tools.

For example, if you **freeze** the shoulder of a character, and use the Pose Move tool to move its hand, dynamic posing works in a similar manner to **IK posing**. You adjust the position of the hand, and the elbow and wrist joint angles are automatically updated to reflect this position.

Alternatively, if you select all the arm mass objects of a character, and use the Pose Rotate tool to rotate its shoulder, dynamic posing works in a similar manner to FK posing. You adjust the angle of the shoulder, and the arm objects downstream of the shoulder are effectively **locked** together and move as a single unit.

Moving objects with Pose Move

You will commonly want to pose joints and mass objects in characters, including in the environment character, by using the Pose Move tool.

Posing the environment character

When you use the Pose Move tool to pose mass objects in the **environment** character, you are effectively moving its mass objects around in a similar manner to using the Move tool. The main difference is addition of collision object penetration avoidance.

Posing other characters

When you use the Pose Move tool to pose other characters, you are effectively changing the joint angles of the character to specify a pose. You can use the Pose Move tool with joints or with mass objects, with the same result. You are not actually editing the character definition itself in any way. For example, you are not actually moving mass objects or collision objects with respect to their parent joints.

To activate the Pose Move tool:

- To activate the Pose Move tool, select **Edit > Pose Move**. The keyboard shortcut is **A**. Alternatively, you can click the **Pose Move** button  in the Main Toolbar. When the Pose Move tool is active, it will be displayed in the Main Toolbar using the **selection colour**.

To select objects using the Pose Move tool:

- You can only select one object at a time when using the Pose Move tool. If you try and select multiple objects, only the last object will be pose moved.
- Only joints and mass objects may be posed using the Pose Move tool. If you try and select a collision object, its parent mass object will be selected instead.
- You can select individual joints and mass objects in any character. This is different to the behaviour of the Move tool. If you try and select an individual joint or mass object with the Move tool active, the entire character is selected instead.

Using the Pose Move tool

The **Pose Move** tool works in a similar manner to the **Move** tool.

You can use the Pose Move tool to freely move a joint or mass object in the virtual world.

You can also use the Pose Move tool to move an object along a **specific direction** or in a **specific plane**. This is called a **constrained move**. You can perform constrained moves along any of the **global** X, Y or Z directions, or in any of the global XY, YZ or XZ planes. You can also perform constrained moves in the **local** X, Y or Z directions, or in the local XY, YZ or XZ planes of the first selected object.

To pose move an object in any direction:

- To move the selected object freely in space, select inside the **purple square** manipulator and drag the mouse. The entire Pose Move tool manipulator is displayed using a yellow highlight colour when you are freely moving selected objects.

To pose move an object in a specific direction:

- To move the selected object along the **X direction**, select the **red** arrow manipulator and drag the mouse.
- To move the selected object along the **Y direction**, select the **green** arrow manipulator and drag the mouse.
- To move the selected object along the **Z direction**, select the **blue** arrow manipulator and drag the mouse.

When you are performing a constrained move along a particular direction, the corresponding arrow manipulator is displayed using a yellow highlight colour. Make sure correct arrow is highlighted before starting to drag.

To pose move an object on a specific plane:

- To move the selected object in the **XY plane**, select the segment of the outer **light blue (aqua)** box manipulator between the **red** and **green** arrow manipulators and drag the mouse.
- To move the selected object in the **YZ plane**, select the segment of the outer **light blue (aqua)** box manipulator between the **green** and **blue** arrow manipulators and drag the mouse.
- To move the selected object in the **XZ plane**, select the segment of the outer **light blue (aqua)** box manipulator between the **red** and **green** arrow manipulators and drag the mouse.

When you are performing a constrained move in the particular plane, the two corresponding arrow manipulators that define the plane are displayed using a yellow highlight colour. Make sure correct arrows are highlighted before starting to drag.

To change the coordinate system of a constrained pose move:

- By default, the Pose Move tool is aligned along the global coordinate system.
- To toggle the Pose Move tool coordinate system so that it is aligned with the local coordinate system of the first selected object, select **Edit > Toggle Coordinate System**. The keyboard shortcut is **X**. The Pose Move tool manipulator updates to reflect the active coordinate system.
- When you change the coordinate system for the Pose Move tool, this setting becomes the default for subsequent uses of the Pose Move tool. It also becomes the default for the **Pose Rotate** tool.

To cancel a pose move:

- A pose move can be cancelled at any time while you are dragging the Pose Move tool manipulator.
- To cancel a pose move, **right-click**. This moves the Pose Move tool manipulator back to its original position at the start of the move.

Rotating objects with Pose Rotate

You will commonly want to pose joints and mass objects in characters, including in the environment character, by using the Pose Rotate tool.

Posing the environment character

When you use the Pose Rotate tool to pose mass objects in the **environment** character, you are effectively rotating its mass objects around in a similar manner to using the Rotate tool. The main difference is addition of collision object penetration avoidance.

Posing other characters

When you use the Pose Rotate tool to pose other characters, you are effectively changing the joint angles of the character to specify a pose. You can use the Pose Rotate tool with joints or with mass objects, with the same result. You are not actually editing the character definition itself in any way. For example, you are not actually moving mass objects or collision objects with respect to their parent joints.

To activate the Pose Rotate tool:

- To activate the Pose Rotate tool, select **Edit > Pose Rotate**. The keyboard shortcut is **S**. Alternatively, you can click the **Pose Rotate** button  in the Main Toolbar. When the Pose Rotate tool is active, it will be displayed in the Main Toolbar using the **selection colour**.

To select objects using the Pose Rotate tool:

- You can only select one object at a time when using the Pose Rotate tool. If you try and select multiple objects, only the last object will be pose rotated.
- Only joints and mass objects may be posed using the Pose Rotate tool. If you try and select a collision object, its parent mass object will be selected instead.
- You can select individual joints and mass objects in any character. This is different to the behaviour of the Rotate tool. If you try and select an individual joint or mass object with the Rotate tool active, the entire character is selected instead.

Using the Pose Rotate tool

The **Pose Rotate** tool works in a similar manner to the **Rotate** tool.

You can use the **Pose Rotate** tool to freely rotate one or more objects in the virtual world.

You can also use the Pose Rotate tool to rotate objects in about a **specific axis or plane**. This is called a **constrained rotate**. You can perform constrained rotations around any of the **global** X, Y or Z directions. You can also perform constrained rotations around any of the **local** X, Y or Z directions of the first selected object. Additionally, you can perform a constrained rotation in the **viewport plane**.

One difference between the Pose Rotate tool and the Rotate tool is that when using the Rotate tool, you can temporarily change the rotation pivot point position. The nature of dynamic posing means that there is no equivalent concept in Pose Rotation.

To pose rotate an object in any direction:

3. Rotating objects freely in any direction of often called **trackball rotation**. Select somewhere in the region **near the centre** of the manipulator. Be careful to avoid any of the specific manipulator circle controls. If you select too far from the centre of the manipulator, you will enter box selection mode.
4. Drag the mouse to freely rotate the selected entities. During trackball rotation, a grey filled circle is displayed in the interior of the Pose Rotate tool manipulator.

To pose rotate an object around a specific axis or plane:

- To rotate the selected objects around the **X axis**, select the **red** circle manipulator and drag the mouse.
- To rotate the selected objects around the **Y axis**, select the **green** circle manipulator and drag the mouse.
- To rotate the selected objects around the **Z axis**, select the **blue** circle manipulator and drag the mouse.
- To rotate the selected objects in the **viewport plane**, select the outer **light blue (aqua)** circle manipulator and drag the mouse.

When you are performing a constrained drag around a particular axis or in a particular plane, the corresponding circle manipulator is displayed using a yellow highlight colour. Make sure correct circle manipulator is highlighted before starting to drag.

To change the coordinate system of a constrained pose rotate:

- By default, the Pose Rotate tool is aligned along the global coordinate system.
- To toggle the Pose Rotate tool coordinate system so that it is aligned with the local coordinate system of the first selected object, select **Edit > Toggle Coordinate System**. The keyboard shortcut is **X**. The Pose Rotate tool manipulator updates to reflect the active coordinate system.
- When you change the coordinate system for the Pose Rotate tool, this setting becomes the default for subsequent uses of the Pose Rotate tool. It also becomes the default for the **Pose Move** tool.

To cancel a pose rotation:

- A pose rotation can be cancelled at any time while you are dragging the Pose Rotate tool manipulator.
- To cancel a pose rotation, **right-click**. This moves the Pose Rotate tool manipulator back to its original position at the start of the rotation.

Resetting poses

Each character (other than the environment character) has a **default pose**. For the **standard simulation character**, the default pose is a **T-pose**. When you create custom characters, you can specify a different default pose.

When you are posing a character, you can **reset** its pose. This reverts the character back to its default pose. It does not change the position or orientation of the character itself.

To reset the pose:

1. **Select** one or more characters. You can select entire characters, or any of their child objects. Keep in mind that Reset Pose does not apply to the Environment character.
2. Select **Character > Reset Pose** to reset the pose of all selected characters.

   Alternatively, Click the **Reset Pose** button  in the Main Toolbar. There is no keyboard shortcut for this command.

   Note Prior to *endorphin* 2.6, the **Pose Move** or **Pose Rotate** tools had to be active before you were able to reset character poses. This restriction no longer applies in *endorphin* 2.6.

Saving and loading poses

After you have edited a character pose, you may want to **reuse** the pose again in the scene, or in other scenes. *endorphin* allows you to save a pose to an external file, and load this pose file onto other characters in the same scene, or onto characters in other scenes.

Poses are stored in binary pose files with **.nma** extensions.

To save a character pose:

1. Select the character. You can use the viewport, or the Node View, or the Timeline Editor, to select the character.
2. Select **Character > Save Pose…**. The **Save As** dialog appears. Browse to a folder in which to save the pose, or select an existing **.nma** pose file to overwrite.
3. Click **Save** to save the pose to an **.nma** file.

To load a character pose:

1. Move the time slider in the Timeline Editor to the frame in which you are going to add the pose.
2. Select the character that you wish to pose. You can use the viewport, or the Node View, or the Timeline Editor, to select the character.
3. Select **Character > Load Pose…**. The **Open** dialog appears. Browse to an **.nma** pose file that contains the pose that you want to use.
4. The **Load Pose** dialog appears. This dialog allows you to specify whether or not to apply the pose to the character cube, and/or to the character root joint, and includes separate options for the position (**Use Translation**) and orientation (**Use Orientation**) of the pose.

You will turn these settings on when you want to use the global position and orientation of the character pose. If you want to keep your existing character position and orientation—and only adjust its relative pose—then you will need to turn these settings off. These settings are on by default.

5. Click **OK** to load this pose onto the selected character. This will replace the current character pose.
6. When you load a pose from a file, no **pose keyframe** is created. If you want to create a keyframe at that frame, you will need to do so manually by right-clicking at the desired frame position in the Timeline Editor and selecting **Create Single Key**.

Sample poses

endorphin ships with some example poses. These are installed into the folder **Resources\Poses\Custom**, and include a poses such as crouching, rolling and holding weapons.

Freezing and locking

When you use **dynamic pose tools** such as Pose Move and Pose Rotate to pose a character, you are effectively using physical dynamics to manipulate the character. The character is effectively suspended in a weightless environment, with dynamic posing operations applying **forces** to the character in order to achieve the required pose move or pose rotate.

Dynamic pose tools can only be used on one joint or mass object at a time. However, the **effect** of using these tools applies to the **entire** character. For example, if you select the hand of the standard simulation character and pose move it vertically upwards, the entire character will be dragged along behind it, since the entire character is a single articulated structure in the virtual world.

There are two main ways to constrain a dynamic pose: **freezing** mass objects and **locking** mass objects. Freezing and locking *only* apply to mass objects. Use **freezing** when you want to work with the pose using an inverse kinematics (IK) approach. Use **locking** when you want to work with the pose using a forward kinematics (FK) approach.

To pose in an IK manner by freezing mass objects:

1. Ensure that either the **Pose Move** or **Pose Rotate** tool is active.
2. **Right-click** on a mass object to **freeze** it, and unfreeze all other mass objects. Hold down **Ctrl** and **right-click** mass objects to toggle the frozen state of each mass object without affecting any other mass object. This allows you to freeze multiple mass objects. Frozen mass objects are displayed using the grey **frozen object colour**.

3. When a mass object is frozen, it effectively divides the articulated structure of the character joint hierarchy into two. When you manipulate joints or mass objects on one side of this divide, no joints on the other side of mass object are affected.

   For example, if you freeze the **LeftShoulderMass** mass object of the standard simulation character, and use the Pose Move or Pose Rotate tools to manipulate any of its arm mass objects, you can effectively pose the arm in an IK manner without affecting any other part of the character.

4. The frozen state of each mass object is preserved, even when the dynamic pose tools are deactivated. When the Pose Move or Pose Rotate tools are next reactivated, the previous frozen state of each mass object will be restored. However, the frozen state of each mass object is not cleared each time the scene is reopened.

To pose in an FK manner by locking mass objects:

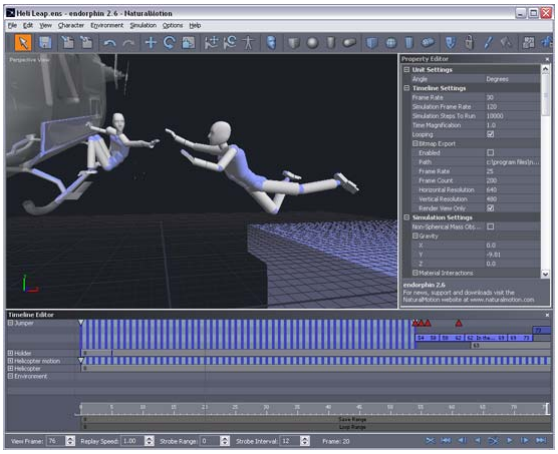
1. Ensure that either the **Pose Move** or **Pose Rotate** tool is active.
2. **Click** on a mass object to select it. This is the mass object that is to be moved or rotated.
3. Hold down the **Ctrl** key and click on other mass objects to cumulatively select them. Effectively you are locking these additional mass objects to the original mass object.
4. With the set of mass objects selected, use the Pose Move or Pose Rotate tools. The **entire** set of locked mass objects moves together with the original mass object.

   For example, if you select the **LeftUpperArmMass**, **LeftForearmMass** and **LeftForearmTwistMass** mass objects of the standard simulation character, you can use the Pose Rotate tool to rotate the LeftUpperArmMass mass object, and rotate the arm as a single, locked object.

What are *endorphin* keyframes?

You can create **keyframes** for each character in an *endorphin* scene.

Keyframes work in a similar, if more basic manner to other animation systems. Keyframes store information about the position and orientations of the joints of a particular character at a particular frame. You can identify **static** keyframes in the Timeline Editor as vertical lines drawn in the **dark blue keyframe colour**. You can create keyframes for all characters, including the Environment character.



Keyframes and simulations

When a character is **not** dynamically simulated—that is, its simulation mode is any of the No Simulation, Collision Only or Collision With Momentum settings—its motion is controlled by its keyframes. If a particular frame has a keyframe, that keyframe controls the positions and orientations of the joints and mass objects of the character. If a particular frame does not have a keyframe, *endorphin* **linearly interpolates** positions and orientations using the nearest keyframes.

When a character is dynamically simulated—that is, its simulation mode is Full Simulation—its static keyframes are **ignored**. Instead, characters are affected by timeline events such as forces, constraints, active poses and behaviours, and also physical processes such as gravity, friction and collisions.

Animation events

Animation events are **containers** of keyframes that have been created by importing animation data from an FBX, XSI, BVH, AMC or Vicon V file. The keyframes in an animation event **override** static keyframes. You cannot edit individual keyframes in an animation event. However, you can **clip** the event so that only a subset of the keyframes are used.

Working with keyframes

There are a number of ways to add and remove **keyframes**. We commonly refer to the keyframes that belong directly to the character as **static keyframes**. Static keyframes appear on the character timeline as vertical blue lines.

Characters may also have animation events applied to their timelines. Animation events themselves are composed of keyframes. However, unlike static keyframes, you can globally move and rotate the keyframes stored in animation events.

To create a new static keyframe by posing:

- When you use the **Pose Move** or **Pose Rotate** tools to pose a character, a keyframe is **automatically** created for that character at the specified frame. Keep in mind that you can use the Pose Move and Pose Rotate tools to manipulate joints or mass objects, but not to manipulate collision objects.

To create a new static keyframe:

- To create a new static keyframe for a specific character, **right-click** on the character timeline and select **Create Key**. Alternatively, **select** the character in the Timeline Editor, Node View or one of the viewports, and select **Edit > Create Key**. The keyboard shortcut is **B**. Keep in mind that you **must** select the character before creating the key, unless you are using the character timeline context menu.
- The new key is created at the position of the **time slider**. The key is **not** created at the location of the right-click (if you are using the character timeline context menu). You cannot create keyframes for frames that do not have a corresponding buffered frame. In practice, this means that if you want to create a keyframe on, say, frame 100, you must first simulate at least 100 frames before you can create that keyframe.

To create new static keyframes for all frames:

- When you save an endorphin scene, buffered frames are not saved into the scene file. However, all keyframes are saved into the scene file. A useful way to permanently store all the data in the frame buffer is to create a corresponding keyframe at every frame, using the data in the frame buffer. This is often called **baking** the animation.
- To bake keys for every frame for a specific character, right-click on the character timeline and select **Create All Keys** or **All Keys With Simulation Events**. Choose the latter command if you want *endorphin* to create additional simulation events at the first keyframe and the last keyframe.
- To bake keys for every frame for every character, right-click on any character timeline and select **Create All Keys (All Characters)** or **All Keys With**

Simulation Events (All Characters). Choose the latter command if you want *endorphin* to create additional simulation events at the first keyframe and the last keyframe for every character.

To delete a static keyframe:

- **Right-click** on the character timeline on the **exact** keyframe that you want to delete, and select **Delete Key At Frame**. The context menu will display the frame number that you have selected. For example, if you have right-clicked on frame 65, the corresponding command will be **Delete Key At Frame 65**.

To delete all static keyframes:

- **Right-click** on the character timeline and select **Delete All Keys**. A confirmation dialog will appear asking you to confirm that you want to delete all the keyframes.

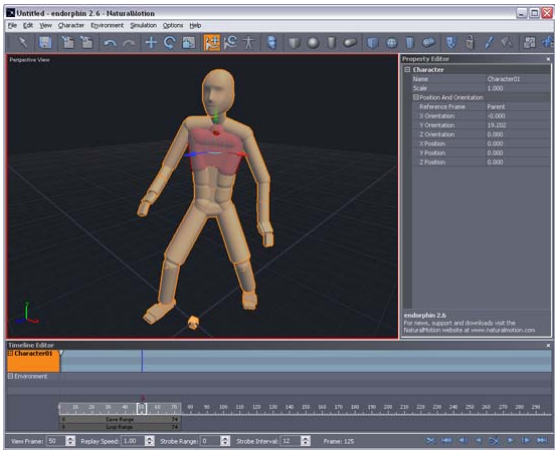
To delete a set of static keyframes:

- **Right-click** on the character timeline and select **Delete Key Range…**. A **Range** dialog appears with Start and End edit boxes. Specify the start frame and end frame, and click **OK** to delete the corresponding range of keyframes.
- **Right-click** on the character timeline and select **Delete Keys From Frame**. The context menu will display the frame number that you have selected. For example, if you have right-clicked on frame 65, the corresponding command will be **Delete Keys From Frame 65**. A confirmation dialog will appear asking you to confirm that you want to delete the specified keyframes.

Using keyframing to animate characters

You will generally simulate most characters in an *endorphin* scene using either dynamic simulation or by driving them using animation keyframes stored in animation events.

In some cases, you may find it useful to animate some characters using traditional keyframing. For example, you may have a vehicle in your scene that you want to animate using keyframes, rather than via an animation event. You can also use keyframing to animate mass objects of the Environment character.



To animate a character using keyframing:

1. If the character is not already in the scene, add it using the **Add Character** command. Position the character using the **Move** and **Rotate** tools so that it is not penetrating any other character already in the scene. Static keyframes are stored in character space, which means that when you move the character, all it's static keyframes will move with it.

   You can also keyframe mass objects of the **Environment** character. Create any mass objects you want to animate, and set their required properties. Keep in mind that keyframed mass objects are not affected by dynamics, so settings such as size and density can be ignored.

2. Run a simulation for as many frames as you need in order to create the keyframes that you require. For example, if you want keyframes on frames 0, 20, 85 and 115, then you will need to simulate for at least 115 frames. You can ignore the actual results of the simulation—the important aspect of this step is to fill the frame buffer. This is because you can only create keyframes in *endorphin* where there is a corresponding buffered frame.

3. Move the time slider to the frame at which you want to create your keyframe.

4. Use the **Pose Move** and **Pose Rotate** tools to move and rotate the joints and mass objects of the character in order to generate the desired pose. A keyframe will automatically be created whenever you use the Pose Move or Pose Rotate commands. Do not use the Move or Rotate tools to when creating or editing keyframes. If a keyframe already exists, it will be updated each time you use the Pose Move or Pose Rotate tools.

5. If you want to create other keyframes, move the time slider to each of those desired frames, and use the Pose Move and Pose Rotate tools at each frame to create a corresponding keyframe.

6. Add a simulation event to the character timeline, and set the simulation mode to either No Simulation, Collision Only or Collision With Momentum.

7. Move the simulation event to the first frame that you want to be driven by keyframing. Usually this will be frame 0, but it can be any frame. Keep in mind that any keyframes prior to this simulation event will be ignored. When you simulate, the character's motion will use the static keyframes, and will be interpolated between those keyframes at frames with no specified keyframe.

Chapter 7

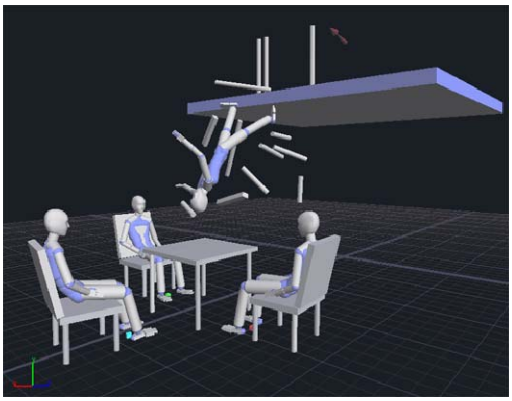
Simulations

What is an *endorphin* simulation?

A **simulation** is the process of allowing the virtual world to evolve over time. Simulations are used in *endorphin* to generate new animation data.

Objects in the virtual world may be **animated** or **simulated**. The motion of animated objects is controlled by keyframes, whereas the motion of simulated objects depends upon physical processes such as forces and collisions.

Simulation characters are articulated sets of objects that can employ **behaviours** to interact with the environment—and with other characters—in an autonomous and intelligent way.



Introducing the frame buffer

When a simulation is running, it generates a sequence of **animation frames** that are stored in a **frame buffer**. Frames begin with **frame 0**, which is also called the **start frame**.

Each time a new simulation is run, the frame buffer from any previous simulation is cleared. After a simulation has been run, you can **replay** the animation in the frame buffer.

When a scene is saved, animation frames in the frame buffer are **not** saved. This is because they can be readily regenerated by running the simulation again. *endorphin* is **deterministic**, which means that you can be sure that running a simulation again will

produce the same resulting frame buffer data, provided that the virtual world description has not changed.

You can export animation data in the frame buffer into a variety of standard animation formats, such as FBX, BVH and XSI.

Running simulations

The **Simulation Toolbar** is located in the lower-right corner of the *endorphin* application frame. This toolbar contains a set of buttons to allow you to start, stop and replay simulations. These commands are also available from the Menu Bar. Many of these commands also have keyboard shortcuts.

Simulations attempt to run in **real time**. This means that *endorphin* will try and display a simulation so that the rate of time passing in the virtual world matches the rate of time passing in the real world. When scenes become quite large, the simulation rate may be slower than real time.

Multiple viewports and simulation rates

endorphin simulations typically run more slowly when you are displaying **multiple** viewports. It is good practice to toggle back to **single** viewport display before starting a new simulation. Note that **playback** rates are unaffected by the use of multiple viewports.

To run a simulation:

- Select **Simulation > Simulate** to run a new simulation. The keyboard shortcut is **Ctrl+Spacebar**. Alternatively, click the **Simulate** button  in the Simulation Toolbar.
- When a simulation is running, the label **Simulating…** appears in each viewport, and the active viewport frame is displayed using the red **active viewport colour**. Also, the Simulate button in the Simulation Toolbar is replaced by the **Stop** button  , and is highlighted using the orange highlight colour.
- When you run a simulation, any existing frames in the frame buffer are cleared.
- When you are running a simulation, the Timeline Editor does not prevent operations such as moving, editing or deleting timeline events. However, we recommend that you do **not** attempt to modify the timeline while a simulation is running.

To stop a simulation:

- Select **Simulation > Play/Stop** to stop a running simulation. The keyboard shortcut is **Spacebar**. Alternatively, click the **Simulate** button  in the Simulation Toolbar.
- If you do not stop a simulation, it will stop automatically when the maximum number of simulation frames has been generated.

To clear the frame buffer:

- Select **Simulation > Clear Frames** to clear the frame buffer. Alternatively, click the **Clear Frames** button  in the Simulation Toolbar. There is no keyboard shortcut for this command.
- When you clear the frame buffer, the time slider is reset to the **start frame**.

To change the simulation frame rate:

1. Ensure no entities are selected.
2. In the **Timeline Settings** properties in the Property Editor, locate **Simulation Frame Rate**. The simulation frame rate specifies how many discrete physics engine steps to run per second of virtual world time. The higher the number of steps, the more accurate the simulation. The default simulation frame rate is 120 steps per second.
3. You can edit this setting. Note that this is a system-level setting, rather than a scene-level setting.

To change the storage frame rate:

1. Ensure no entities are selected.
2. In the **Timeline Settings** properties in the Property Editor, locate **Frame Rate**. The frame rate specifies how frequently to store frames into the frame buffer. The default frame rate is set to 60 frames per second. The frame rate should not be higher than the simulation frame rate.
3. You can edit this setting. Note that this is a system-level setting, rather than a scene-level setting.

To change the maximum simulation timesteps:

1. Ensure no entities are selected.
2. In the **Timeline Settings** properties in the Property Editor, locate **Simulation Steps To Run**. This specifies the maximum number of simulation steps to run.

The default setting is 10000 simulation steps. You can always manually stop a simulation before this number of timesteps has been reached.

3. You can edit this setting. Note that this is a system-level setting, rather than a scene-level setting. Also note that this is the maximum number of **simulation** steps, rather than the maximum number of frames stored in the frame buffer.

Intersecting characters

New characters are always created at the **origin** of the virtual world. This can sometimes be confusing, as it means that you can inadvertently add multiple characters to a scene at the same position. It is good practice to always **move** new characters away from the origin of the virtual world **before** adding further characters.

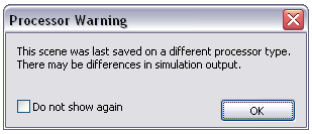
If you inadvertently attempt to simulate with multiple intersecting characters in your scene, the simulation rate will fall rapidly as the characters involved violently decouple. If this occurs, stop the simulation and manually separate the intersecting characters. Likewise, it is important to ensure that no characters are intersecting with any of the **Environment** character collision objects before running a simulation.

Using different processors

When you run simulations multiple times using the same scene, *endorphin* will always generate the **same** resulting simulation. When you save a scene, the frame buffer itself is not stored in the scene. This is because you can reload the scene and resimulate it in order to regenerate the same animation frame buffer.

Sometimes, however, you may find slight differences between the simulations on different computers, if they use different processor types. That is, simulations are always consistent for a given processor, but may vary slightly between different processors. For example, a given scene may simulate slightly differently on an Intel Pentium processor to an AMD Athlon processor. These differences are usually quite subtle.

endorphin stores the processor type when it saves a scene. If you open the scene on a computer with a different processor type, the **Processor Warning** dialog appears. This dialog is used to indicate that the scene may simulate slightly differently to when it was last opened. You can prevent this dialog from reappearing for that scene by checking the **Do not show again** checkbox.



Replaying simulations

Once you have simulated a scene and have animation stored in the frame buffer, you can review the animation by **replaying** it. Replaying animation from the frame buffer is much faster than regenerating it using a simulation, particularly for larger or more complex scenes. You can always replay in **real time**, regardless of how fast or slow the scene was to simulate.

To replay a simulation:

- Select **Simulation > Play/Stop** to replay a simulation. The keyboard shortcut is **Spacebar**. Alternatively, click the **Play** button in the Simulation toolbar. When a simulation is being replayed, the Play button in the Simulation Toolbar is replaced by the **Stop** button, and is highlighted using the orange highlight colour.
- Select **Simulation > Play Backwards** to replay a simulation in reverse. The keyboard shortcut is **Alt+Spacebar**. Alternatively, click the **Play Backwards** button in the Simulation toolbar. When a simulation is being replayed backwards, the Play Backwards button in the Simulation Toolbar is replaced by the **Stop** button, and is highlighted using the orange highlight colour. Replaying simulations in reverse can sometimes be a useful technique when assessing animation quality.

To stop a replay:

- Select **Simulation > Play/Stop** to stop a replay of the simulation. The keyboard shortcut is **Spacebar**. Alternatively, click the **Play** button in the Simulation Toolbar (or if you are replaying backwards, click the **Play Backwards** button in the Simulation Toolbar).

To change the replay speed:

1. Ensure no entities are selected.
2. In the **Timeline Settings** properties in the Property Editor, locate **Time Magnification**. This setting specifies the replay speed. For example, a setting of 1.0 implies playback at real time, whereas a setting of 2.0 implies a 50% slow motion replay. The default time magnification is 1.0. Slow motion replays is a very useful technique when assessing animation quality.

3. You can edit this setting. Note that this is a system-level setting, rather than a scene-level setting.
4. You can also edit this setting using the **Replay Speed** setting in the Timeline Editor. Keep in mind that the Replay Speed in the Timeline Editor is the inverse of the Time Magnification setting in the Timeline settings. For example, a time magnification of 2.0 will be displayed as a replay speed of 0.5.

To change the looping setting:

1. Ensure no entities are selected.
2. In the **Timeline Settings** properties in the Property Editor, locate **Looping**. This setting specifies whether to continually repeat the replay when the last frame in the frame buffer is reached. Looping is turned on by default.
3. You can edit this setting. Note that this is a system-level setting, rather than a scene-level setting.

Navigating frame-by-frame

Once you have simulated a scene and have animation stored in the frame buffer, you can review the animation by **replaying** it. Alternatively, you can manually navigate through the frame buffer in a frame-by-frame manner. This is very useful for when you need to fine-tune a simulation and carefully assess the resulting animation.

To display the first frame:

- Select **Simulation > Go To First Frame** to display the first frame in the frame buffer. The keyboard shortcut is **Ctrl+LeftArrow**. Alternatively, click the **Go To First Frame** button  in the Simulation toolbar.

To display the previous frame:

- Select **Simulation > Go To Previous Frame** to display the previous frame in the frame buffer. The keyboard shortcut is **LeftArrow**. Alternatively, click the **Go To Previous Frame** button  in the Simulation toolbar.

To display the next frame:

- Select **Simulation > Go To Next Frame** to display the next frame in the frame buffer. The keyboard shortcut is **RightArrow**. Alternatively, click the **Go To Next Frame** button  in the Simulation toolbar.

To display the last frame:

- Select **Simulation > Go To Last Frame** to display the last frame in the frame buffer. The keyboard shortcut is **Ctrl+RightArrow**. Alternatively, click the **Go To Last Frame** button  in the Simulation toolbar.

To display a specific frame:

- Enter a specific frame number in the **View Frame** text box in the Timeline Editor, or use the spinner button to change the frame number.
- Alternatively, select and the **time slider** in the Timeline Editor and drag it to the desired frame. The process of manually replaying certain sections of the frame buffer by dragging the time slider is often called **scrubbing the timeline**.
- Alternatively, **double-click** on the Timeline Editor **buffered frames marker** to display the corresponding frame.

Camera tracking

When you are simulating a scene, or replaying a simulation, you may be interested in the motion of a particular object or character. If that object or character is moving significantly through virtual world space during the simulation, it can be difficult to assess its motion.

To make it easier to follow the motion of objects that are moving through the scene, you can **camera track** specific objects in the scene. When camera tracking is turned on, the relative distance and direction from the viewport camera to the tracked object is maintained.

Camera tracking is turned off by default whenever you open a scene. You can camera track in both perspective and orthogonal viewports.

To turn on camera tracking:

- To turn on camera tracking, select **View > Camera Tracking**. Alternatively, click the **Camera Tracking** button  in the Main Toolbar. The keyboard shortcut is **Ctrl+T**. When camera tracking is active, the Camera Tracking button in the Main Toolbar has an orange **selection colour** background. To turn off camera

tracking, deselect **View > Camera Tracking**. Alternatively, click the **Camera Tracking** button in the Main Toolbar again, or press the keyboard shortcut again.

To select a camera tracking object:

- When camera tracking is on, the camera tracks any object that is selected. You can track any kind of object in the virtual world. Usually you will only camera track a single object, but if multiple objects are selected then camera tracking will operate by tracking the **midpoint** of the selected objects.
- The tracked object always moves to the centre of the active viewport. When camera tracking is on, selecting different objects can cause large changes in the viewport display. If you find this disconcerting or awkward, you may want to turn camera tracking off **before** selecting the tracked object, and then turn camera tracking back on.
- Camera tracking only applies to the active viewport. If you are displaying multiple viewports, camera tracking only occurs in the active viewport. The active viewport can be identified by its yellow **active viewport border**. Click in any viewport to activate it.
- If no objects in the scene are selected, the camera will not be tracked, even if camera tracking is on.
- If there are one or more objects in the scene selected, the camera will be tracked. If will track during simulations, during replay of simulations, and even when navigating through the frame buffer manually using the mouse or keyboard.

Simulation keyboard shortcuts

Using keyboard shortcuts for running and replaying simulations is an efficient way to work with *endorphin*.

Summary of keyboard shortcuts

- **Ctrl+Spacebar** to start simulating, and **Spacebar** to stop simulating.
- **Spacebar** to start replaying, and **Spacebar** again to stop replaying.
- **Alt+Spacebar** to start replaying backwards, and **Spacebar** to stop replaying backwards.
- **LeftArrow** and **RightArrow** to display to the previous frame and next frame.
- **Ctrl+LeftArrow** and **Ctrl+RightArrow** to display the first frame and last frame.

Introducing the Timeline Editor

The **Timeline Editor** displays time-based scene data.

Each character in the scene has a separate character **timeline**, made up of multiple timeline **event tracks**. You can add **events** to a character timeline to influence the simulation in various ways.

When you simulate a scene, the **frame buffer** is filled with animation frame data. The Timeline Editor displays the frame buffer, along with a **time slider** that lets you navigate through the frame buffer. The time slider and frame buffer are both displayed over the **time ruler**, which displays frame numbers.

The Timeline Editor also includes markers that represent various ranges. The **loop range** specifies the frames that are replayed when you are replaying the simulated frames. The **save range** specifies the frames that are exported when you export simulated frames to an external animation file. The **strobe range** specifies the frames that are visualised by the strobing tool.

Changing the display frame

There are various ways you can use the Timeline Editor to change which frame is being displayed.

To identify what frame you are hovering over:

- When you move the mouse over the Timeline Editor, a black vertical **frame identifier** line follows the mouse cursor. The frame identifier line makes it easier to identify the corresponding frame associated with the mouse position. The corresponding frame number is displayed in the status bar alongside the Simulation Toolbar.

To display a specific frame:

- **Double-click** on the **buffered frames marker** to display the corresponding frame. The buffered frames marker is drawn in light grey along the time ruler, below the event tracks.
- Alternatively, select and the **time slider** and drag it to the desired frame. The time slider is the white rectangular manipulator located on the buffered frames marker. The **[x]** in the centre of the time slider indicates the frame it is associated with. The process of manually replaying certain sections of the frame buffer by dragging the time slider is often called **scrubbing the timeline**. You cannot drag the time slider beyond the limit of buffered frames.

- Alternatively, enter a specific frame number in the **View Frame** text box in the Timeline Editor, or use the spinner button to change the frame number.
- Alternatively, if you have **event markers** on an event track, you can **double-click** the event marker. If the event is a single-frame event, that frame is displayed. If the event is a multi-frame event, the frame corresponding to the location of the double-click is displayed.
- You can also use the **LeftArrow** and **RightArrow** keyboard shortcuts to display to the previous frame or next frame.
- You can also use the **Ctrl+LeftArrow** and **Ctrl+RightArrow** keyboard shortcuts to display the first frame and last frame.

Note If you are dragging the time slider towards the start frame, you might find that you cannot drag any further with the active frame not yet at frame 0. This is due to the width of the time slider manipulator. To drag the time slider to frame 0, it is important that you select the time slider manipulator somewhere between its centre of the manipulator and its right-hand edge. If you select the manipulator somewhere between its centre and its left-hand edge, you may find the lowest frame you can drag to is frame 1 or frame 2.

Panning and zooming

There are various ways you can change which frames are visible in the Timeline Editor.

To pan to a different start frame:

- By default, the time ruler shows the **start frame**, which is **frame 0**.
- Hold down **Alt** and drag anywhere in the Timeline Editor with the **left mouse button** to **pan** left or right in the Timeline Editor. This changes the start frame in the time ruler, and is useful if you are working with events that start a considerable time after the start frame.

To zoom to a different frame range:

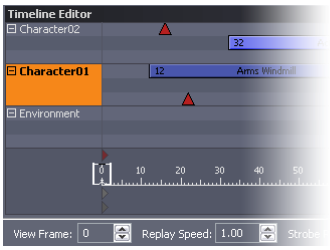
- By default, the time ruler shows **300** frames for new scenes.
- Hold down **Alt** and drag anywhere in the Timeline Editor with the **right mouse button** to **zoom** left or right in the Timeline Editor. This changes the number of frames visible in the time ruler. Zooming out is useful if you want to work with events that are located after frame 300. Zooming in is useful is you have complex series of events that require detailed editing.

To zoom to fit all items:

- Right-click anywhere in the Timeline Editor and select **Zoom To Fit**. This resets the time ruler so that it shows the start frame. It also zooms the time ruler so that all event markers, and all buffered frames, are visible. This is a useful command if you need to quickly review the entire timeline.

Selecting characters

You can **select** characters using the Timeline Editor. This can be useful, as all the characters in the scene can be quickly accessed in the Timeline Editor, whereas they might be distributed over a large area in the virtual world and difficult to locate. Also, you can use the Timeline Editor to select hidden characters, and also to select the Environment character.

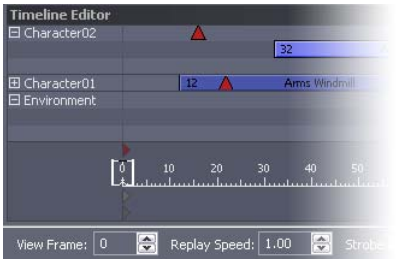


To select characters:

- To select a single character using the Timeline Editor, **click** anywhere inside the character timeline to the **left** of the vertical divider that separates the character name from the timeline tracks themselves.
- To **additively** select characters using the Timeline Editor, hold down the **Ctrl** key and select the character in the timeline. This selects the character and but does not affect any existing selection.
- The **Select** tool does not need to be active in order to select characters using the Timeline Editor.
- When characters are selected, their character timeline uses the orange **selection colour** as a background colour.

Expanding and collapsing characters

When there are many characters in a scene, the Timeline Editor can get quite large. To keep the Timeline Editor to a manageable size, it can be useful to **collapse** character timelines down to a single event track. This does not affect the simulation itself in any way. *endorphin* preserves the expanded or collapsed state of each character timeline. When a scene is re-opened, these settings are restored.



To expand and collapse individual character timelines in the Timeline Editor:

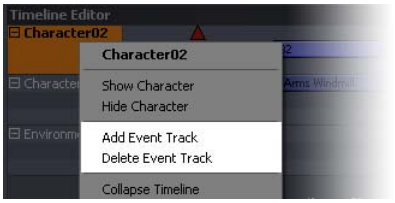
- Click the **[-]** button next to the character name in the Timeline Editor to **collapse** an expanded character timeline. When a character timeline is collapsed, the scene simulates in the same way as before. However, the individual character event tracks are combined into a single event track. The event markers that were previously distributed over multiple character event tracks are now displayed in the same event track.
- Click the **[+]** button next to the character name in the Timeline Editor to **expand** a collapsed character timeline. The event markers that were previously distributed over a single event track are now displayed over multiple event tracks.
- Alternatively, **right-click** to the left of the vertical divider and select **Expand** to expand a collapsed character timeline, or select **Collapse** to collapse an expanded character timeline.

To expand and collapse all character timelines in the Timeline Editor:

- **Right-click** to the left of the vertical divider and select **Expand All** to expand all the character timelines, or select **Collapse All** to collapse all the character timelines. These commands are useful when you have many characters in a scene.

Adding and removing event tracks

When characters are added to a scene, their character timelines have three event tracks by default. You can **add** event tracks to a character timeline, which can be useful is your character timeline has many event markers. Also, you can **remove** event tracks if you have a relatively simple character timeline and want to preserve space by removing unused event tracks.



To add an event track to a character timeline:

- **Right-click** to the left of the vertical divider and select **Add Event Track**. This adds a new event track below any other existing event track.

To remove an event track from a character timeline:

- **Right-click** to the left of the vertical divider and select **Delete Event Track**. This deletes the event track corresponding to the position of the right-click.

  If the event track has no event markers, it is deleted automatically. If the event track has some event markers, a warning dialog is displayed that confirms that all event markers on that event track will be deleted. Click **Yes** to delete the event track and all of its markers. If you do not want to delete the event markers on the track, click **No** and then manually move the event markers on that track to another event track.

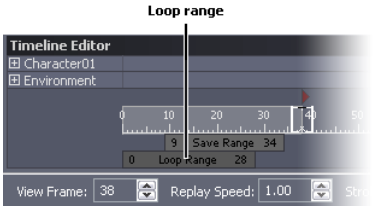
  Characters must have at least one event track. You cannot delete all the event tracks for a given character.

Setting the loop range

The Timeline Editor also includes markers that represent various ranges. The **loop range** specifies the frames that are replayed when you are replaying the simulated frames. The loop range marker is located below the time ruler. The label **Loop Range** is displayed on the marker itself, along with the start frame and end frame.

By default, each time the simulation is stopped, the loop range updates so that it coincides with the number of buffered frames. You can also set a **custom loop range** by manually changing the loop range using the loop range marker.

If you change the loop range manually, it will not longer update each time the simulation is stopped. The custom loop range is saved with a scene and restored when the scene is re-opened.



To set a custom loop range:

- To change the **position** of the loop range, hover over the loop range until the cursor changes to a **hand** cursor. **Click** on the loop range marker and drag left or right. The number of frames in the loop range remains unchanged.
- To change the **start frame** of the loop range, hover over the **left** edge of the loop range until the cursor changes to a **left-right arrow** cursor. **Click** on the loop range marker drag left or right. This changes the start frame and the number of frames in the loop range.
- To change the **end frame** of the loop range, hover over the **right** edge of the loop range until the cursor changes to a **left-right arrow** cursor. **Click** on the loop range marker drag left or right. This changes the end frame and the number of frames in the loop range.

  Note If you adjust the loop range so that the start of the loop range is at a frame beyond the number buffered frames, then replaying the simulated frames will appear to have no effect.

To reset a custom loop range:

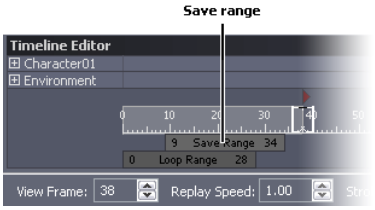
- To **reset** a custom loop range, **right-click** on the loop range marker and select **Reset Range**. Alternatively, **right-click** on the frame buffer and select **Reset Loop Range**. This resets the loop range to coincide with the number of buffered frames. Also, once the loop range has been reset, it will update again each time the simulation is stopped.

Setting the save range

The Timeline Editor also includes markers that represent various ranges. The **save range** specifies the frames that are exported when you export simulated frames to an external animation file. The save range marker is located below the time ruler. The label **Save Range** is displayed on the marker itself, along with the start frame and end frame.

By default, each time the simulation is stopped, the save range updates so that it coincides with the number of buffered frames. You can also set a **custom save range** by manually changing the save range using the save range marker.

If you change the save range manually, it will no longer update each time the simulation is stopped. The custom save range is saved with a scene and restored when the scene is re-opened.



To set a custom save range:

- To change the **position** of the save range, hover over the save range until the cursor changes to a **hand** cursor. **Click** on the save range marker and drag left or right. The number of frames in the save range remains unchanged.
- To change the **start frame** of the save range, hover over the **left** edge of the save range until the cursor changes to a **left-right arrow** cursor. **Click** on the save range marker drag left or right. This changes the start frame and the number of frames in the save range.
- To change the **end frame** of the save range, hover over the **right** edge of the save range until the cursor changes to a **left-right arrow** cursor. **Click** on the save range marker drag left or right. This changes the end frame and the number of frames in the save range.

To reset a custom save range:

- To **reset** a custom save range, **right-click** on the save range marker and select **Reset Range**. Alternatively, **right-click** on the frame buffer and select **Reset Save Range**. This resets the save range to coincide with the number of buffered frames. Also, once the save range has been reset, it will update again each time the simulation is stopped.

Strobing

The **strobe preview** is a useful visualisation tool for previewing simulations. Strobing allows you to view multiple superimposed animation frames, giving you a simulation preview without the need to actually run a simulation.

The **strobe range** is the frame range to be strobed. The start frame of a strobe range is always the current location of the time slider. By default, the strobe range is zero frames. The strobe range marker is displayed in the Timeline Editor as a red triangle marker above the time slider.

The **strobe interval** specifies the interval between strobed frames. For example, suppose the time slider is at frame 100, and the strobe range is 45 frames. This means the potential frames that can be strobed will be in the range 100 to 145. If the strobe interval is 10 frames, then **four** frames will actually be strobed. These will be frames 110, 120, 130 and 140. If the strobe interval is 12 frames, then three frames will actually be strobed. These will be frames 112, 124 and 136.

A common use of strobing is when you are making small adjustments to some part of a scene or timeline event, and want to preview the result of that change. Every time you make a change to the scene, to a timeline event, or to current display frame, the strobe preview is updated.

To display the strobe preview, the display mode should be set to **Shaded View** or **Wireframe View**. The strobe preview is not available in Skeletal View. Only mass objects, collision object and graphical objects are displayed in the strobe view. Joints and joint limits are not displayed.

To set the strobe range:

- If the strobe range is zero, hover over the strobe range marker until the cursor changes to a **left-right arrow** cursor. Click on the strobe range marker and drag it to the right to change the strobe range. When the strobe range is at least one frame, the triangular strobe range marker is replaced by a rectangular marker.
- If the strobe range is non-zero, hover over the **right** edge of the strobe range marker until the cursor changes to a **left-right arrow** cursor. Click on the strobe range marker and drag left or right to change the strobe range.
- Alternatively, you can enter a strobe range value in the **Strobe Range** text box below the Timeline Editor, or use the spinner next to this text box to change the strobe range.

- If you extend the strobe range beyond the number of buffered frames, you will notice that the frame buffer is **automatically** filled to coincide with the last frame of the strobe range.

To set the strobe interval:

- Enter a strobe interval value in the **Strobe Interval** text box below the Timeline Editor, or use the spinner next to this text box to change the strobe interval.
- Experiment with different strobe intervals. The default strobe interval is 12 frames. If the strobe interval is too low, you might find that your system performance starts to be affected.

To show or hide the strobe preview:

- When you are working with strobe previews, it is often useful to be able to quickly turn on or off strobing without having to edit the strobe range. To hide the strobe preview, press **Alt+S**. To show the strobe preview, press **Alt+S** again.

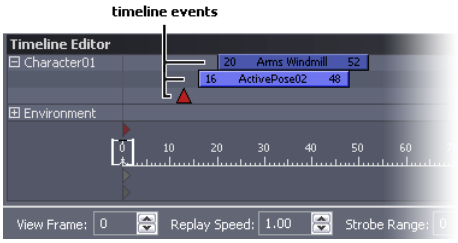
Chapter 8

Timeline Events

Introducing timeline events

When you **edit** a scene in *endorphin*, you are effectively setting up the initial conditions for that scene. If you then simulate the scene, the scene evolves according to standard physical dynamics.

Timeline events allow you to change the way a simulation unfolds. Timeline events apply to particular frames, or ranges of frames, are created and edited using the Timeline Editor. During a simulation, when a frame is encountered that contains as one or more timeline events, these events are **triggered** and their effect is included in the simulated virtual world for that timeslice. You can use events to modify the simulation in subtle or extensive ways.



Event types applying to simulation characters

- **Behaviour events** allow you to specify a particular behaviour for a character, such as **jumping**, **tackling**, **standing** or **reaching**. Behaviour events allow characters to respond intelligently to their environment and to other characters. Behaviour events can only be applied to the **standard simulation character**, and to custom characters that have been derived from the standard simulation character.

Event types applying to all characters

- **Active pose events** allow you to specify an **intended pose** for a character over a range of frames. You can specify the pose itself, as well as the strength of the pose and the objects affected by the pose. Active poses are a kind of dynamic keyframe, and are useful when you want to control certain body parts of a character in some detail.

- **Constraint events** allow you to **lock** a set of mass objects to each other over a range of frames. You can specify the type of constraint and which objects are affected. Constraint events are useful when you want to model effects such a character holding an object.
- **Force events** allow you to apply a **force** to a set of mass objects at a particular frame. You can set the direction and strength of this force, and which objects are affected. Force events are useful when you want to model effects such as recoil due to collision with a projectile.
- **Sever events** allow you to **disconnect** a joint in a joint hierarchy at a particular frame. Sever events are useful when you want to model the breaking of an object.

Event types for animation blending

- **Animation events** allow to you to associate animation data with a character, and drive the character using this data as an alternative to dynamic simulation.
- **Motion transfer events** allow to you dynamically transfer motion from one character to another. Usually, one of the characters will be driven by predefined animation, and the other character is driven dynamically using this motion.
- **Simulation events** allow you to change the **simulation mode** of a character from **simulation** to **animation**, and vice versa. Simulation events are useful when you want to use predefined animation to drive a character over some frames, and then use the full simulation to drive the character for other frames.
- **Transition events** allow you to blend between **simulation** and **animation**. Transition events are useful when you want to smoothly blend between simulation and animation.

Creating timeline events

Timeline events apply to **characters**. Most event types are available to all characters. Some event types, such as behaviour events, only apply to characters derived from the standard simulation character. The environment character also supports a more limited set of event types.

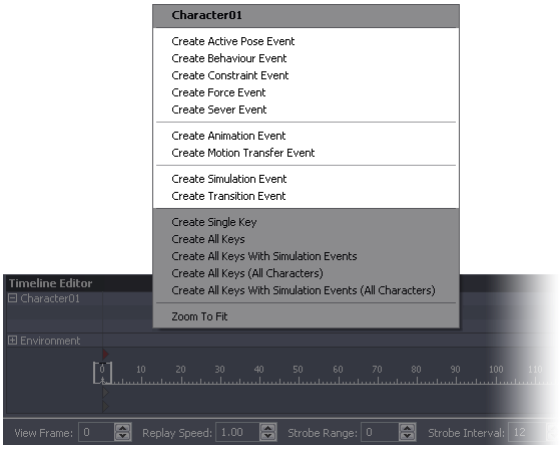
Some event types, such as force, sever and simulation events, are **single-frame** events. Other event types are **multi-frame** events. When you create a timeline event, a corresponding **timeline event marker** is created in the Timeline Editor. The default duration of multi-frame events is 60 frames.

Multi-frame event markers are **rectangular**, and display the name of the event and the start and end frames of the event on the marker. Single-frame event markers have different shapes, depending on the event type. Force events use triangular markers. Simulation events use inverted triangle markers. Sever events use diamond markers.

To create a timeline event using the Timeline Editor:

- **Right-click** in the Timeline Editor on the timeline of the target character, and select **Create Force Event** to create a force event for the specified character. The character does not need to be selected. There are similar commands in the context menu to create active pose, behaviour, constraint, sever, simulation, transition and motion transfer events. If any of these commands are unavailable in the context menu, this means that the corresponding character does not support that event type.

  The new timeline event is created by default at the frame corresponding to the location of the right-click.



To create a timeline event using the Menu Bar:

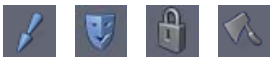
1. Select a character using the viewports, or the Timeline Editor, or the Node View.
2. Select **Character > Force Event** to create a force event for the selected character. There are similar commands in the Menu Bar to create active pose, behavior, constraint, sever, simulation, transition and motion transfer events. There are no keyboard shortcuts for these commands. If a command is not active, this means that the selected character does not support that event type.

   The new timeline event is created by default at **frame 0**.

   If no character is selected, some event types will be created for the environment character. These event types are: force, constraint, simulation and transition events.

To create a timeline event using the Main Toolbar:

1. Select a character using the viewports, or the Timeline Editor, or the Node View.
2. Click the **Force Event** button to create a force event for the selected character. There are similar buttons in the Main Toolbar to create behaviour, constraint and sever events. If the command has no effect, this means that the selected character does not support that event type.

   The new timeline event is created by default at **frame 0**.

   If no character is selected, some event types will be created for the environment character. These event types are: force, constraint, simulation and transition events.

Selecting timeline events

You can **select** timeline event markers using the Timeline Editor. Some timeline events, such as force events, have a corresponding graphic in the viewports. However, most timeline events can only be selected in the Timeline Editor.

To select timeline events:

- To **select** timeline event markers individually, **click** on an event marker in the Timeline Editor. The event marker is selected, and all other entities in the scene are deselected.
- To **box select** timeline event markers, click in the Timeline Editor and **drag**. When you release the mouse, any event markers fully enclosed or partially

enclosed by the selection rectangle are selected, and all other entities in the scene are deselected.

- To additively select timeline event markers, hold down the **Ctrl** key and select or box select timeline events.
- When event markers are selected, they are drawn using the orange **selection colour**.

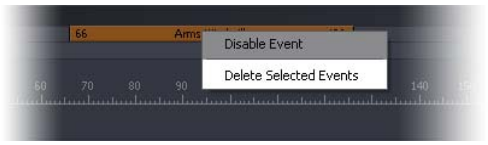
Deleting timeline events

You can **delete** timeline event markers using the Timeline Editor.

To delete timeline events:

1. Select the timeline events that you want to delete. If you want to delete multiple events, hold down the **Ctrl** key and select event markers, or use box selection.
2. Select **Edit > Delete** the selected events. The keyboard shortcut is **Delete**.

   Alternatively, **right-click** any of the selected events and select **Delete Selected Events** to delete the selected events.



Moving and resizing timeline events

You can **move** and **resize** timeline event markers using the Timeline Editor. You can also change the start frame, end frame and frame properties of timeline event markers using the Property Editor.

Changing the event start frame, end frame or frame is a command that can be **undone**.

To move a timeline event:

- **Click** on the timeline event marker and drag left or right. Note that you cannot drag a timeline event marker to a different character.

To move multiple timeline events:

1. Hold down the **Ctrl** key and select the event markers that you want to move. Alternatively, use box selection to select the event markers that you want to

move. You can select multiple event markers on the same character, or across different characters.

2. **Click** any of the selected event markers and drag left or right.

To resize a multi-frame timeline events:

1. Hover over the left or right edges of the timeline event marker until the mouse cursor changes to the **left-right arrow** cursor.
2. **Click** and drag left or right to resize the event.

To change timeline event frames using the Property Editor:

1. **Click** on a timeline event to select it.
2. If the event is a **single-frame** event, locate the **Frame** property in the Property Editor. You can edit this value to change the frame at which the event occurs.

   If the event is a **multi-frame** event, locate the **Start Frame** and **End Frame** properties in the Property Editor. You can edit these values to change the start frame and end frame over which the event occurs.

Changing timeline event priorities

The **priority** of a timeline event corresponds to the index of the timeline track on which the event marker is placed. Events placed on the **topmost** track have the **highest** priority, followed by events on the track underneath the top track, and so on.

During a simulation, when a frame is encountered that has an event marker associated with it, the corresponding event is **triggered**. If multiple events occur on the same frame, the sequence in which they are triggered corresponds to their priority. That is, the events are triggered from the **top** track to **bottom** track.

In many cases, the sequence in which events are triggered does not affect the simulation. In other cases, the sequence **does** affect the simulation. For example, suppose that on a given frame, a character has a **force** event applied, and also has a **constraint** event which begins at that frame. If the force event is triggered before the constraint event, the simulation will probably have a different outcome than if the constraint is triggered before the force event. This is because the constraint will probably override the effect of the force.

Changing the event priority is a command that can be **undone**.

To change the priority of a timeline event:

- **Click** on the timeline event marker and drag up or down. Note that you can cannot drag a timeline event marker to a different character.

To change the priority of multiple timeline events:

1. Hold down the **Ctrl** key and select the event markers that you want to move. Alternatively, use box selection to select the event markers that you want to move. You can select multiple event markers on the same character, or across different characters.
2. **Click** any of the selected event markers and drag up or down.

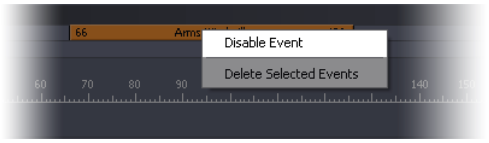
Disabling timeline events

You can **disable** timeline event markers using the Timeline Editor. You can also disable timeline event markers using the Property Editor. By default, new timeline events are enabled. When a timeline event is disabled, it is not triggered during a simulation. Enabling and disabling timeline events allows you to experiment with different event configurations without the need to continually create and delete the same event.

Disabled timeline events are displayed using the grey **disabled colour**. You can select, move, edit and delete timeline events regardless of whether they are enabled or disabled.

To enable or disable a timeline event:

- **Right-click** on the timeline event marker and select **Disable Event** to disable events that are currently enabled, or select **Enable Event** to enable events that are currently disabled.



To enable or disable a timeline event using the Property Editor:

1. **Click** on a timeline event to select it.
2. Locate the **Enabled** property in the Property Editor. You can edit this value to enable or disable the event.

Renaming timeline events

You can **rename** most timeline event markers using the Property Editor.

By default, new timeline events are created with unique names of the form **Force01**, **Constraint02**, and so on. Multi-frame event markers display their name on the marker itself.

You can rename timeline events to give them more meaningful names. For example, you might rename a force event as **Gunshot**, or rename a constraint event as **Hold Railing**. Timeline events must have **unique** names. If you attempt to name multiple events with the same name, *endorphin* will add a numeric suffix of the form 01, 02 and so on, to ensure the name is unique.

Note You cannot rename **behaviour** or **animation** events. The name of a behaviour event always corresponds to the type of behaviour, such as **Arms windmill 2.0**. The name of an animation event always corresponds to the original filename, such as **Jog.fbx**.

To rename a timeline event using the Property Editor:

1. **Click** on a timeline event to select it.
2. Locate the **Name** property in the Property Editor. You can edit this value to change the event name.

Cutting, copying and pasting timeline events

You can **cut**, **copy** and **paste** timeline event markers using the Timeline Editor.

To cut timeline events:

1. Select the timeline events that you want to cut. If you want to cut multiple events, hold down the **Ctrl** key and select event markers, or use box selection. You can select events from multiple characters.
2. Select **Edit > Cut** to cut the selected events. The keyboard shortcut is **Ctrl+X**. The selected events are deleted from the Timeline Editor and copied into the Windows clipboard.

To copy timeline events:

1. Select the timeline events that you want to cut. If you want to copy multiple events, hold down the **Ctrl** key and select event markers, or use box selection. You can select events from multiple characters.
2. Select **Edit > Copy** to copy the selected events. The keyboard shortcut is **Ctrl+C**. The selected events are copied into the Windows clipboard.

To paste timeline events:

- Select **Edit > Paste** to paste events from the Windows clipboard. The keyboard shortcut is **Ctrl+V**. The new events are created on the same characters from which they were cut or copied.

  Most of the properties of the original events are copied when the new events are created. However, some properties are **reset** in the new event. For example, if you cut-and-paste or copy-and-paste behaviour events, the pasted behaviour events have default behaviour variables. They do **not** use the behaviour variables from the source events.

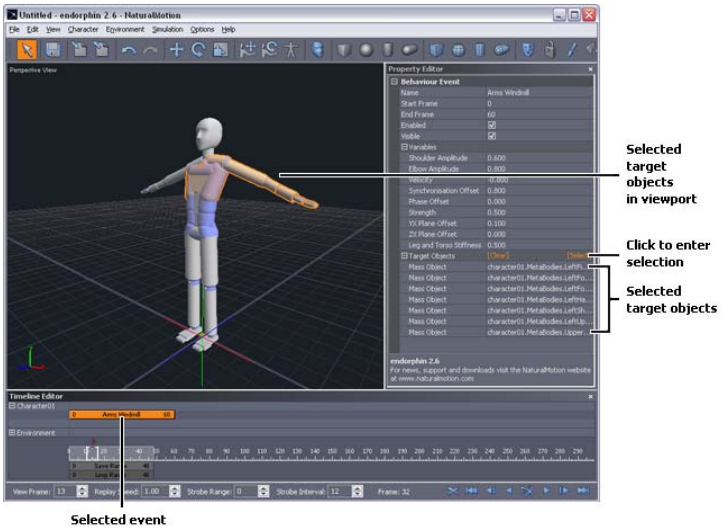
Selecting timeline event target objects

Timeline events are always associated with one of the characters in the scene. Some event types apply to the character as a whole, whereas other event types apply to certain parts of the character. The set of character objects to which an event applies is the **target object set**.

Some events have a target object set that you will rarely need to change. In other cases, such as force events, constraint events and sever events, you will almost always edit the target object set.

When you select a timeline event marker, the target object set for that event is displayed in the viewports as the **dependent selection**, using the dark orange **dependent selection colour**. This allows you to quickly identify the objects in the character that will be affected by the event.

You can edit the target object set using **Selection mode**.



To edit the target object set using Selection mode:

1. Select the timeline event that you want to edit.
2. Locate the property in the Property Editor that has a **[Select]** command hotlink. For behaviour, constraint, force and sever events, this is the **Target Objects** property. For active pose and animation events, this is the **Target Joints** property.
3. Click the **[Select]** command hotlink to enter **Selection mode**. The label **Selecting…** appears in the viewports, and the cursor changes to the **selection hand** cursor.
4. In Selection mode, target objects are displayed using the dark red **dynamic selection colour**.

   Click on an object to clear the target object set and assign the selected object to the target object set.

   Hold down the **Ctrl** key and **click** on objects to add or remove them from the target object set.

   Some event types allow you to select objects from different characters. Force and events can have target objects from **different** characters. Behaviour, active pose, sever and animation events must have target objects from their corresponding character only.

   For behaviour, constraint, force and sever events, the target objects are **mass objects**. If you select a collision object or graphical object, the corresponding parent mass object is used as the target object.

For active pose and animation events, the target object are **joints**. If you select a mass object, collision object or graphical object, the corresponding parent joint is used as the target object.

5. To **exit** Selection mode, **right-click** the viewport. The keyboard shortcut is **Esc**. Alternatively, **click** in the viewport away from any scene object, or **click** anywhere in the Timeline Editor.

Notes

- Behaviour events have a set of target mass objects. By default, behaviour events apply to every mass object. As for other event types, you can change the set of behaviour target mass objects. For behaviour events, if the **Target Objects** set is empty, this indicates that the **entire** character is affected by the event. In contrast, for other event types, if the **Target Objects** or **Target Joints** lists are empty, this indicates that the event does not affect any object. Similarly, the **[Clear]** command hotlink in the behaviour event Property Editor clears the set of target mass objects, which indicates that the **entire** character if affected by the event.
- Motion transfer events have a **Source Character** property which has a **[Select]** command hotlink. This command allows you specify the motion transfer event source character using Selection mode.
- Simulation events have a **Target Object** property which has a **[Select]** command hotlink. However, this hotlink currently does not operate. The target object for a simulation event is always the character to which the event applies.
- Transition events also have a list of **Target Joints**, but these are not currently editable. This functionality may be available in future versions of *endorphin*.

Editing timeline event properties

All timeline events have **properties**.

Editing properties using the Property Editor

When you select an event marker, the properties of the corresponding event are displayed in the **Property Editor**.

The **Name** and **Enabled** properties are common to all event types. The **Frame** property is common to all single-frame event types, such as force, sever and simulation events. The **Start Frame** and **End Frame** properties are common to all multi-frame event types, such as active pose, behaviour, constraint, animation, transition and motion transfer events.

Editing properties using the Timeline Editor

Some event properties can be edited in the Timeline Editor.

You can edit the **Enabled** property for all timeline event markers by right-clicking the marker and selecting **Enable Event** or **Disable Event**. This allows you to quickly change the configuration of timeline events without having to browse to the Enabled property in the Property Editor.

You can horizontally drag all timeline events, and resize multi-frame timeline events, using the mouse. This effectively edits the **Frame** property of single-frame events and the **Start Frame** and **End Frame** properties of multi-frame events.

You can vertically drag all timeline events to a different timeline track using the mouse. This effectively edits the **Priority** property. This property is not displayed in the corresponding Property Editor.

Chapter 9

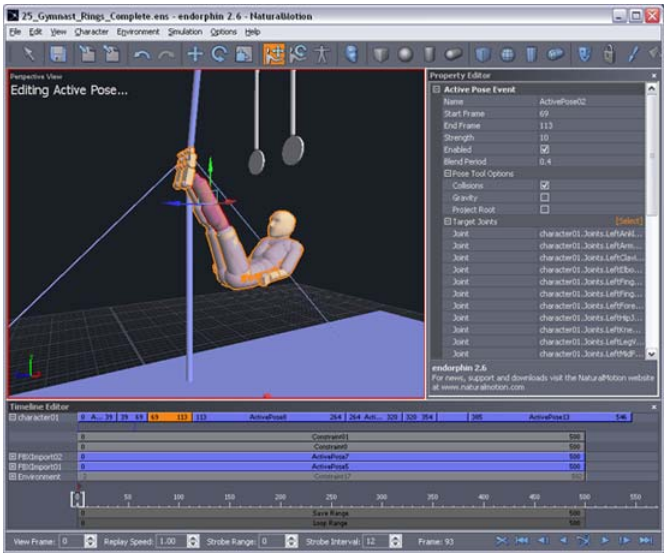
Active Pose Events

What are active pose events?

Active posing is a way of influencing a dynamic simulation using **dynamic keyframing**.

An active pose event allows you to specify a **desired** or **intended** pose for a given character over a range of frames. You can specify a pose for the entire character or for some portion of the character. Depending on the **strength** of the active pose, you can specify to what extent the active pose influences the simulation.

Active pose events are drawn using an indigo (blue-purple) **rectangle marker** in the timeline.



Using active pose events

Active poses are useful when you want to guide or influence the **pose** of a character during a dynamic simulation. Active poses are edited using the **Pose Move** and **Pose Rotate** tools.

You can specify the **strength** of the effect. If the active pose strength is high, the motion of the character will be strongly influenced by the pose. If the active pose strength is low, it will have a more subtle effect on the motion. However, in all cases, the active pose represents a desired or intended pose. The character may or may not be able to actually attain the pose. This will depend on many factors, such as the strength of the pose, the duration of the active pose event, the effect of gravity, forces, constraints and other timeline events, and collisions with other characters and the environment.

You can specify the **target joints** of an active pose. In some cases, you might want the entire joint hierarchy of a character to be influenced by an active pose. For example, if you have a character falling or hanging relatively freely, you may want to use active poses to drive the entire character. In other cases, you might to use an animation event or behaviour event to drive most of a character joint hierarchy, and use active poses over a subset of the character, such as the arm joints, for a specific effect.

Keep in mind when using active poses that these are **local** poses. They can affect the joint angles of the target character, but they cannot move or rotate the target character relative to the virtual world. For example, suppose you add an active pose event to a target character that is tumbling through space in free-fall. The active pose will change the character pose as it falls, but will have no effect on the overall movement and rotation of the character as it falls.

Editing active poses

You can edit the pose used by an active pose event using **Edit Active Pose mode**.

By default, the pose used by an active pose is the default pose for that character. For the standard simulation character, this is a T-pose. You can edit this pose using the **Pose Move** and **Pose Rotate** tools.

The target character is displayed in the pose used by the active pose event. The target joints of the active pose event are displayed using the orange **selection colour**, and the corresponding child mass objects and collision objects of the target joints are displayed using the darker orange **dependent selection colour**.

To edit the pose used by an active pose event:

1. When you create a new active pose event, or select an existing active pose event, you will automatically enter **Edit Active Pose** mode. The label **Editing Active Pose…** appears in the viewports, and the **Pose Move** tool is activated.

2. To edit the pose, use the **Pose Move**  and **Pose Rotate**  tools to edit the pose. All the functionality associated with the Pose Move and Pose Rotate tools, such as locking and freezing mass objects, is available when editing an active pose. Do **not** use the Move, Rotate or Scale tools.

   You can edit the entire pose of the character. However, keep in mind that only the joints in the target joint set will be affected by the pose.

   If you want to reset the pose back to the default pose, **click** the **Reset Pose** button  in the Main Toolbar.

3. To **exit** Edit Active Pose mode, **deselect** the active pose event in the Timeline Editor. They keyboard shortcut is **Esc**.

Saving and loading active poses

After you have edited a character active pose, you may want to **reuse** the pose again in the scene, or in other scenes. *endorphin* allows you to save an active pose to an external file, and load this pose file onto other characters in the same scene, or onto characters in other scenes. You can use a pose file to set a initial character pose, or to set an active pose.

Poses are stored in binary pose files with **.nma** extensions.

To save the pose of an active pose event:

4. **Right-click** on an active pose event marker and select **Save Pose…** to enter Edit Active Pose mode. The **Save As** dialog appears. Browse to a folder in which to save the pose, or select an existing **.nma** pose file to overwrite.

5. Click **Save** to save the active pose to an **.nma** file.

6. Press **Esc** to exit Edit Active Pose mode.

To load a pose onto an active pose event:

7. **Right-click** on an active pose event marker and select **Load Pose…** to enter Edit Active Pose mode. The **Open** dialog appears. Browse to an **.nma** pose file that contains the pose that you want to use for the active pose event.

8. Click **Open** to load this pose onto the selected active pose event. This will replace the current pose used by that event.

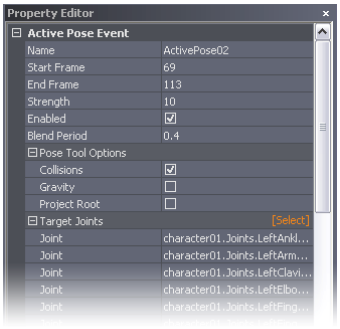
9. Press **Esc** to exit Edit Active Pose mode.

Sample poses

endorphin ships with some example poses that can be used in active pose events. These are installed into the folder **Resources\Poses\Custom**, and include a poses such as crouching, rolling and holding weapons.

Active pose event properties

Force events have **Name** and **Enabled** properties, which are common to all event types, and **Start Frame** and **End Frame** properties, which are common to all multi-frame event types. In addition, constraint events also have the following properties:



Strength

Range: Unlimited positive value. **Default value:** 1.0.

Specifies the **degree** to which the active pose influences the movement of the target joints of the character.

Strengths are dimensionless values. Use large strengths (in the range 50.0 to 100.0) when you want the active pose to be a **dominant** influence on the motion of the character. Use small strengths (in the range 1.0 to 5.0) when you want the active pose to be a **subtle** influence on the motion of the character. Use zero strength to effectively disable the active pose event.

Blend period

Range: Unlimited positive value. **Default value:** 0.0.

Specifies the **time** in seconds over which the active pose event will reach its full strength.

Use a zero blend period when you want the active pose to be immediately applied with its full strength. Use non-zero values when you want to gradually increase the strength of an active pose from zero strength up to its maximum strength as defined by the **Strength** property. For example, a value of 1.0 indicates that the active pose will take one second to reach its full strength. Using blend periods can help make your dynamic simulation **smoother**, and the affect of the active pose less disruptive.

Pose tool: Use collisions

Range: On/Off. **Default value:** On.

Specifies whether to consider **collisions** between the target character and other characters when editing an active pose using the **dynamic posing** Pose Move and Pose Rotate tools. When this option is turned off, you are able to interpenetrate the character with other characters when editing an active pose.

This option is not available when using Pose Move and Pose Rotate to edit poses **except** in Edit Active Pose mode. Active poses represent an **influence** on a character during the course of a simulation, regardless of where the target character is physically located. When you are editing an active pose, the actual position of the character during Edit Active Pose mode is irrelevant, and so collisions between the character and other characters can be safely ignored. It can be useful to off Use Collisions if you find that collisions between the character and other characters are making it difficult to attain the desired pose.

Pose tool: Use gravity

Range: On/Off. **Default value:** Off.

Specifies whether to **gravity** should affect the target character when it is being edited in Edit Active Pose mode.

It can be useful to turn on Use Gravity when you want to edit an active pose in which a character is lying on the ground plane or over some objects on the ground.

Pose tool: Project root

Range: On/Off. **Default value:** Off.

Specifies whether the **position** and **orientation** of the pose character in Edit Active Pose mode should reflect the position and orientation of the target character at the **start frame** (frame 0), or at the **current display frame** as defined by the time slider.

When you edit an active pose, it is important to remember that it is the **relative** positions of the joints that are relevant. The actual position and orientation of the pose character in Edit Active Pose mode are purely for visualisation, and are not used by the active pose event in any way.

When you enter Edit Active Pose mode with the Project Root setting turned off, the position and orientation of the pose character are set to reflect the position and orientation of target character as they were defined in the **start frame** (frame 0).

However, when you enter Edit Active Pose mode with the Project Root setting turned **on**, the position and orientation of the pose character are set to reflect the position and orientation of target character as they are defined in the **frame buffer** for the **current frame**. This can be useful when you want to examine a simulation to assess how much a character has been influenced by an active pose.

Target objects

Range: Set of joints. **Default value:** All joints.

Specifies the set of joints to which the active pose applies.

By default, an active pose event will apply to all the joints in the corresponding character, except the **root** joint. At least one joint must be in this set. All the joints must be selected from the character to which the active pose event applied. You cannot edit the joint names directly. Instead, click the **[Select]** command hotlink and edit the target object set in **Selection** mode.

Chapter 10

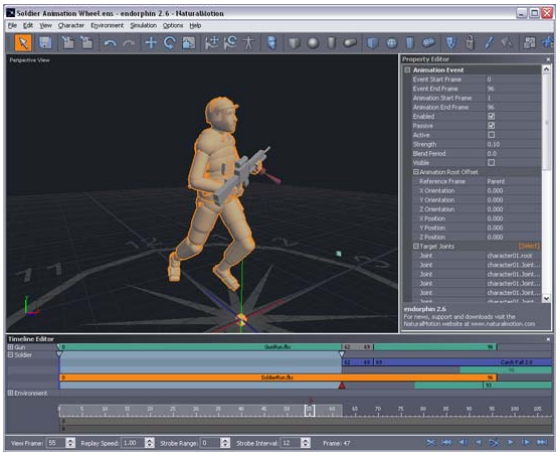
Animation Events

What are animation events?

Animation events allow you to drive characters using animation data. This allows you to create much more complex scenes by using *endorphin* dynamic simulations in conjunction with animation that you may have already generated by motion capture or by traditional keyframing in other animation systems.

You can set the start and end frames of an animation event, and specify which joints are affected. Animation events can be applied to any character. You can apply animation events to prop characters, such as vehicles, as well as to simulation characters.

Animation events are drawn using a light green **rectangle marker** in the timeline. The marker displays the name of the source animation file. Unlike most other event types, you cannot rename animation events.



Using animation events

Animation events are containers for a series of **animation keyframes** that are applied to a character.

You can create animation events by **importing** animation contained in FBX, BVH, XSI, AMC or Vicon V files onto a character that is already in the scene. You can also import animation data directly into the scene, which creates a new character as well as an animation event.

Once animation had been added to a character in the form of an animation event, you can **move** and **rotate** it using viewports or the Property Editor. You can also change its start and end frames by moving its marker in the Timeline Editor, or by using the Property Editor.

You can add multiple animation events to the same character. Usually, these events will be placed at different frames throughout the timeline. If multiple animation events overlap on a range of frames, the vertical **priority** of the animation event will determine which event will be used for those frames and for any common joints. That is, the highest event will take precedence over any lower events.

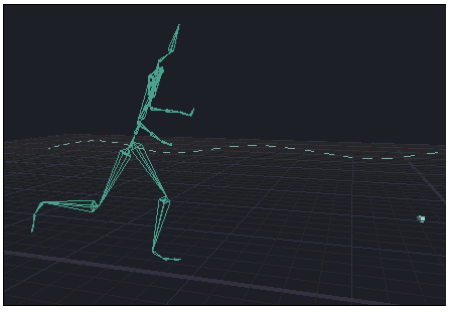
Simulation and transition events

Animation events are nearly always used in conjunction with **simulation** events and **transition** events. These events are used to change the **simulation mode**. When the simulation mode of a character is No Simulation, Collision Only or Collision With Momentum, the character is fully driven by animation data; when the simulation mode of a character is Full Simulation, the animation data can be used to drive the muscles of the character but the rest of the simulation will also influence the character's motion.

- At the **start** of an animation event, you will typically use a **transition** event to create a smooth blend into the animation event. At the end of the transition event, the simulation mode will either be No Simulation, Collision Only or Collision With Momentum, and the character will be fully driven by the keyframed data stored in the animation event.
- At the **end** of an animation event, you will typically use a **simulation** event to set the simulation mode back to Full Simulation. This will result in instantaneous return to dynamic simulation, with velocity and momentum continuity for each mass object of the character.

Animation events in the viewport

Animation events are displayed in viewports. You can select, move and rotate animation events by interacting with the corresponding animation event graphics in the viewport.



To display animation events:

- Select the animation event in the Timeline Editor, and turn on the **Visible** property of the event in the Property Editor.
- Animation event data is drawn as a joint hierarchy using the same light green colour as used by the animation event marker in the Timeline Editor. In addition, the **motion trail** of the root node of the animation data hierarchy is drawn as a dashed line. The motion trail is a useful way to visualise the overall trajectory of the animation data, regardless of the specific frame being displayed by the time slider.

To select animation events:

- You can **click** on an any part of the animation data graphic in a viewport, including the motion trail, to select the corresponding animation event.

  You can also use **box selection** to select animation data graphics.

To move animation events:

- You can use the **Move** tool to select and move animation data graphics in the viewport.

  Moving animation data graphics effectively changes the **X Position**, **Y Position** and **Z Position** properties of the animation root offset of the corresponding animation event.

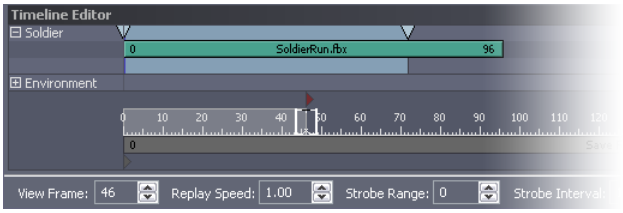
To rotate animation events:

- You can use the **Rotate** tool to select and rotate animation data graphics in the viewport.

Rotating animation data graphics effectively changes the **X Orientation**, **Y Orientation** and **Z Orientation** properties of the animation root offset of the corresponding animation event.

Animation events in the Timeline Editor

When you import animation data from a source FBX, BVH, XSI, AMC or Vicon V file, every frame of data in the file is stored as a separate keyframe in the animation event. You cannot edit individual keyframes in the animation event.



Animation range and event range

The **animation range** of an animation event represents the entire set of keyframes stored in the animation event. The Animation Start Frame and Animation End Frame properties of the event define this range. The light-green rectangular animation marker is drawn to indicate the animation range.

The **event range** of an animation event represents the **subset** of keyframes that are actually used. The Event Start Frame and Event End Frame properties of the event define this range. The darker bevelled rectangular border of the animation marker is drawn to indicate the event range. Also, the start frame and end frame of the event range are displayed on the animation marker.

By default, the event range matches the animation range. However, you can reduce the event range so that it is smaller than the animation range. This is called **clipping** the animation event. Increasing the event range so that is larger than the animation range can result in no keyframes being available to be used by the event when simulating over these frames.

To move an animation event:

1. Select the animation event in the Timeline Editor, or in one the viewports.
2. **Click** on the animation event marker in the Timeline Editor and drag left or right. This changes the frame at which the animation event occurs. Alternatively, you can edit the Animation Start Frame or Animation End Frame properties directly in the Property Editor. Moving the animation event does not affect any clipping applied to the event.

Note You will frequently use simulation and transition events in conjunction with animation events. When moving animation events, keep in mind that you will usually need to move any corresponding simulation or transition events as well. A useful technique is to multiple select all the simulation, transition and animation events that form a single unit before moving the animation event. This will ensure that when moving any of these events, all the selected events move together. You can use **box selection** to select multiple timeline events. Alternatively, you can press **Ctrl** and multiple select timeline events.

To clip the start of an animation event:

1. Select the animation event in the Timeline Editor, or in one of the viewports.
2. Hover over the **left** edge of animation event marker in the Timeline Editor until the cursor changes to the **left-right arrow** cursor. Click and drag left or right to change the Event Start Frame. Alternatively, you can edit the Event Start Frame property directly in the Property Editor.

   When you move the Event Start Frame to a frame after the Animation Start Frame, you effectively **clip** the frames between the Animation Start Frame and Event Start Frame. These frames will not be used by the animation event. Moving the Event Start Frame to a frame **before** the Animation Start Frame can result in no keyframes being available to be used by the event when simulating over these frames.

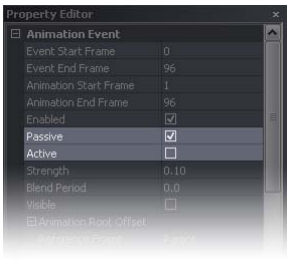
To clip the end of an animation event:

1. Select the animation event in the Timeline Editor, or in one the viewports.
2. Hover over the **right** edge of animation event marker in the Timeline Editor until the cursor changes to the **left-right arrow** cursor. Click and drag left or right to change the Event End Frame. Alternatively, you can edit the Event End Frame property directly in the Property Editor.

   When you move the Event End Frame to a frame before the Animation End Frame, you effectively **clip** the frames between the Event End Frame and Animation End Frame. These frames will not be used by the animation event. Moving the Event End Frame to a frame **after** the Animation End Frame can result in no keyframes being available to be used by the event when simulating over these frames.

Passive and active animation

Animation events are containers for a series of **animation keyframes** that are applied to a character. There are two ways that the keyframes in an animation event can drive a character—via **passive** animation and via **active** animation.



Passive animation

Most of the time, animation events are used passively. This is the default mode for new animation events. To use an animation event in passive mode, turn on the **Passive** property in the Property Editor.

The character must be in the No Simulation, Collision Only or Collision With Momentum simulation modes for the passive option to have any meaning. If the passive option is enabled (and the active disabled) the animation event will have no influence on the character while in Full Simulation mode.

When an animation event is used passively, the animation keyframes **directly** drive the motion of the corresponding character. As the character is not in Full Simulation mode, it is unaffected by forces, constraints, gravity or any other physical processes. Use this mode when you want the character to be driven entirely by the source animation data.

Active animation

A new animation mode, **active** animation, was introduced in *endorphin* 2.5. To use an animation event in active mode, turn on the **Active** property in the property editor. Animation events can be used in passive mode, or in active mode, or in both passive and active modes.

The character must be in the Full Simulation mode to use active animation. If the character is in the No Simulation, Collision Only or Collision With Momentum simulation modes for any frames, animation keyframes are ignored for those frames (unless the passive option is also turned on).

When an animation event is used actively, the animation keyframes drive the joints of the character in a manner similar to using a succession of **active poses**. These active poses influence the motion of the character as part of dynamic simulation, but also compete with forces, constraints, behaviours, gravity, friction and other physical processes. You can

modify the **Blend Period** and **Strength** properties of the animation event to change how it behaves in active animation mode.

By default, active animation events do not affect the root joint of a character. That is, they affect the individual joints of a character but not its overall position or orientation. For example, if you apply a run cycle animation to a character that is falling under the influence of gravity, the arms and legs of the character will move roughly according to the run cycle, but the character will continue to tumble independently of this motion.

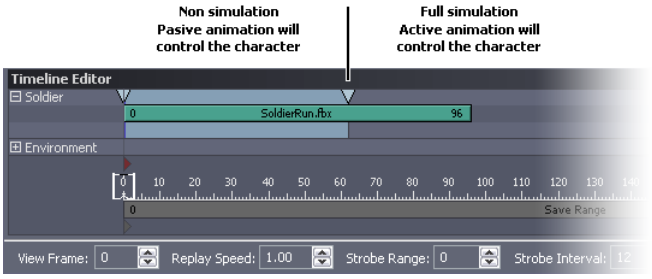
It is possible to control the root by using the Root constraint, in a manner similar to LockWithParent constraints.

Working with passive and active animation

To use an animation event in both active and passive modes, turn on both the **Passive** and **Active** properties in the Property Editor. The event will act as an active animation event whenever the character is in Full Simulation mode, and will act as a passive animation event whenever the character is in No Simulation, Collision Only or Collision With Momentum modes.

Using the same animation event in both passive and active modes is a common technique when you are blending from dynamic simulation to keyframed animation. To do so, you must add **transition** or **simulation** events to the character timeline during the event to change the simulation mode.

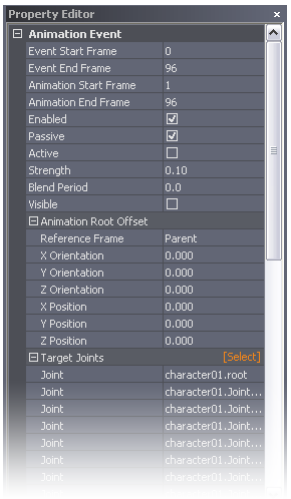
For example, suppose you have imported a run cycle as an animation event between frames 100 and 400, and you want gradually blend from dynamic simulation into pure passive animation. You might add a transition event between, say, frames 140 and 160. Between frames 100 and 140, the animation event acts as an **active** animation event. Between frames 140 and 160, there is a **transition** from active to passive animation. Between frames 160 and 400, the animation event acts as a **passive** animation event. By adjusting the timing of the animation and transition events, you can usually obtain smooth transitions from dynamic simulation to pure passive animation.



Animation event properties

Animation events have an **Enabled** property, which is common to all event types. Animation events do **not** have a Name property. Instead, animation event markers display their animation source filename.

In addition, animation events also have the following properties:



Event start frame

Range: Unlimited positive value. **Default value:** 0.

Specifies the first frame of the animation event in which keyframes are used for active or passive animation.

Any frames between the animation start frame and the event start frame are ignored. For example, if the animation range is between frames 100 and 200, and the event range is between frames 120 and 180, keyframes in the ranges 100 to 119 and 181 to 200 are ignored. Editing the event start frame so that it occurs before the animation start frame can result in no keyframes being available to be used by the event when simulating over these frames.

Event end frame

Range: Unlimited positive value. **Default value:** Number of frames in the source animation file.

Specifies the last frame of the animation event in which keyframes are used for active or passive animation.

Any frames between the event end frame and the animation end frame are ignored. For example, if the animation range is between frames 100 and 200, and the event range is between frames 120 and 180, keyframes in the ranges 100 to 119 and 181 to 200 are ignored. Editing the event end frame so that it occurs after the animation end frame can result in no keyframes being available to be used by the event when simulating over these frames.

Animation start frame

Range: Unlimited positive or negative value. **Default value:** 1.

Specifies the start frame of the event.

Equivalent to the Start Frame property of other multiframe events, except that you cannot change the length of the animation range. Changing the animation start frame also changes the animation end frame. The animation range depends upon the number of frames of animation in the source file.

Note In *endorphin* you will not see the effect of an event on the world until at least one frame after the first frame of the event. Thus, an event that starts on frame 20 has no effect on the world at frame 20. Because of this, the default animation start frame is frame 1 rather than frame 0. You can set the animation start frame to be zero or some negative value, which can be useful if you want to use a later part of the animation at the start of the timeline. However, keep in mind that any animation keyframes before frame 1 are ignored, regardless of the **event** start frame.

Animation end frame

Range: Unlimited positive or negative value. **Default value:** Number of frames in the source animation file.

Specifies the end frame of the event.

Equivalent to the End Frame property of other multiframe events, except that you cannot change the length of the animation range. Changing the animation end frame also changes the animation start frame. The animation range depends upon the number of frames of animation in the source file.

Passive

Range: On/Off. **Default value:** On.

Specifies whether the keyframes in the animation event are used when the simulation mode is No Simulation, Collision Only or Collision With Momentum. Use this setting when you want to drive characters **directly** using animation keyframes.

Active

Range: On/Off. **Default value:** Off.

Specifies whether the keyframes in the animation event are used when the simulation mode is Full Simulation. Use this setting when you want to drive the joints of the character like a succession of **active poses** that are based on the keyframes. These active poses influence the motion of the character as part of dynamic simulation, but also compete with forces, behaviours, gravity and other physical processes.

Strength

Range: 0.0 to 1.0. Default value: 0.1.

Specifies the degree to which the animation keyframes influence the movement of the target joints of the character. Only used in active animation mode.

Strengths are dimensionless values. Use large strengths when you want the active animation to be a dominant influence on the motion of the character. Use small strengths when you want the active animation to be a subtle influence on the motion of the character. Use zero strength to effectively disable the active animation.

Blend period

Range: Unlimited positive value. **Default value:** 0.0.

Specifies the time in seconds over which an active animation event blends the active poses from the pose of the character at the beginning of the event into the poses stored by the event. Only used in active animation mode.

Use a zero blend period when you want the active animation to immediately start using the active poses stored in the event. Use non-zero values when you want to gradually move from using the opening pose as the active pose to using those stored by the event. For example, a value of 1.0 indicates that the active animation will take one second before it's fully using the animations stored in the event. Using blend periods can help make your dynamic simulation smoother.

Visible

Range: On/Off. **Default value:** On.

Specifies whether the animation data is displayed in the viewports.

Root constraint mode

Range: FullHold / FullRotationHold / FullTranslationHold / TranslationAndRotationHold / RotationHold / TranslationHold / NoHold. **Default value:** NoHold.

Specifies how the character root node should be driven during an **active** animation event. This property is only available if the animation event has the **Active** property turned on.

By default, active animation events rotate character joints, but do not move or rotate the character root node itself. That is, active animation is generally used to apply a series of local active poses to a character, rather than to move or rotate the character as a whole. However, you can adjust the root constraint mode so that the character as a whole is moved or rotated by the animation keyframes. The available root constraint modes are:

- **NoHold** will not apply any special movement or rotation to the root node during the duration of the active animation event. This is the default mode.
- **TranslationAndRotationHold** will move and rotate the character root node during the duration of the active animation event using the specified positional and angular strengths as limit strengths.
- **TranslationHold** will move the character root node during the duration of the active animation event using the specified positional strengths as limit strengths.
- **RotationHold** will rotate the character root node during the duration of the active animation event using the specified angular strengths as limit strengths.
- **FullHold** will move and rotate the root to match the animation, with no upper limit of the positional or angular strengths.
- **FullTranslationHold** will move the root to match the animation, with no upper limit of the positional strengths.
- **FullRotationHold** will rotate the root to match the animation, with no upper limit of the rotational strengths.

Animation root offset

Range: Unlimited values for each position and orientation. **Default value:** 0.0 for each position and orientation.

Specifies the position and orientation of an **offset** applied to the animation keyframe data. By default, there is no offset applied. This means that the keyframes stored in the animation event exactly match the data stored in the animation source file.

You can apply a **positional** offset to the keyframe data stored in the animation event by modifying the X Position, Y Position or Z Position properties. These properties can be set in the Property Editor. Alternatively, you can use the **Move** tool in the viewports to directly move the keyframe animation data.

You can apply a **rotational** offset to the keyframe data stored in the animation event by modifying the X Orientation, Y Orientation or Z Orientation properties. These properties can be set in the Property Editor. Alternatively, you can use the **Rotate** tool in the viewports to directly rotate the keyframe animation data.

Note The Reference Frame property is currently unused. The position and orientation values of the offset are always displayed in World coordinates, regardless of the reference frame setting.

Target joints

Range: Set of joints in the corresponding character. **Default value:** All joints in the corresponding character.

Specifies the set of joints to which the animation event applies. At least one joint must be in this set. By default, all the joints in the corresponding character are in the target joint set. You cannot edit the joint names directly. Instead, click the **[Select]** command hotlink and edit the target object set in **Selection** mode.

Chapter 11

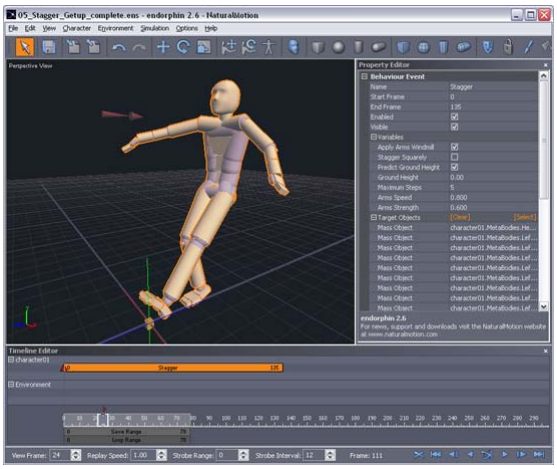
Behaviour Events

What are behaviour events?

Behaviour events allow you to apply adaptive, intelligent motion to a character over a range of frames. Behaviours employ advanced artificial intelligence techniques and allow you to quickly add high-quality humanlike movement to characters in a dynamic simulation.

You can apply behaviours to an entire character, or to a part of a character. Behaviours can only be applied to the **simulation characters**. These are the bipedal, humanoid characters that include the standard simulation character and characters derived from the standard simulation character.

Behaviour events are drawn using a dark blue **rectangle marker** in the timeline.



Using behaviour events

Behaviour events allow you to add high-quality humanlike movement to characters in a dynamic simulation. You can choose the behaviour **type** from dozens of behaviours in the **Behaviour Library**.

There are two broad categories of behaviours—active behaviours and reactive behaviours.

Active behaviours include behaviours such as **Jump** and **Writhe**. When an active behaviour is triggered, it will immediately have an influence on the motion of the corresponding character.

Conditional behaviours include **Stagger** and **Catch Fall**. When a conditional behaviour is triggered, it will have an influence on the character only if the conditions are relevant to that behaviour. For example, a Stagger behaviour will only influence the motion of a character if the character is unbalanced. If the character is balanced, a Stagger behaviour will have no visible effect.

All behaviours have a set of **variables** that allow you to customise the behaviour. For example, the Jump behaviour allows you to specify the strength and timing of the jump.

The **target object set** for a behaviour event can include any number of mass objects from the target character. By default, a behaviour will affect the entire character unless you modify the target object set. Although you apply a sever event to mass objects, it is actually the corresponding parent joint of each target mass object which is driven by the behaviour.

Behaviours can be divided into various sub-categories, such as **whole body** behaviours, **hand and arm** behaviours and **leg** behaviours. In addition, there are some **stiffness and damping** behaviours that affect physical attributes of the character, such as joint stiffness and body damping. These behaviours do not generate intelligent motion in themselves; rather they are used in conjunction with other behaviours to generate specific effects.

Behaviour versions

Behaviours are often upgraded between major releases of *endorphin*. Behaviour names include a numeric extension that specifies the **behaviour version number**. The behaviours released with *endorphin* 1.6 were all Version 1. Most behaviours have been upgraded one or more times since that release.

- In some cases, a newer version of a behaviour replaces an older version. For example, **Arms Windmill 2** replaced **Arms Windmill 1**. Behaviours are improved in various ways, such as adding new behaviour parameters or adding visualization graphics in the viewport.
- In other cases, the newer behaviour is an **alternative** to the previous behaviour, and is shipped alongside the previous behaviour. You are then able to choose between different versions of the behaviour. For example, **Writhe 2** and **Writhe 3** are both available with *endorphin* 2.6. Each version works slightly differently, and

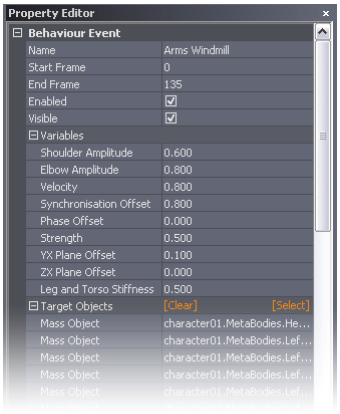
you might find that one or the other works better, depending on the nature of your scene.

The behaviours available for a given simulation character depend upon the internal version of that character. **Upgrading** an older character will usually change the list of available behaviours.

For example, if you open the **Arrows.ens** sample scene and select the **Stagger** behaviour on the **Character01** timeline, you will notice that only **Version 1** behaviours are available. This is because Character01 is based upon the *endorphin* 1.6 standard simulation character. If you upgrade Character01 so that it is based upon the *endorphin* 2.6 standard simulation character, and then select the Stagger behaviour again, you will notice that newer versions of many behaviours are now available. The Stagger1 behaviour appears as a **special case** at the bottom of the Behaviour Library—for backwards compatibility—but you will not be able to add new Stagger1 behaviours to the scene.

Behaviour event properties

Behaviour events have an **Enabled** properties, which is common to all event types, and **Start Frame** and **End Frame** properties, which are common to all multi-frame event types. In addition, behaviour events also have the following properties:



Name

Specifies which **type** of behaviour to use. Select one of the behaviours in the **Behaviour Library**. The Name also includes a numeric **behaviour version number** as a suffix.

Visible

Specifies whether behaviour visualisation is displayed in the viewport. Although all behaviours have this property, not all behaviours currently include these visualisation graphics. Future releases of *endorphin* will include visualisation graphics for a wide range of behaviours. The following behaviours are equipped with behaviour visualisation:

- **Catch Fall:** Ground height visualisation.
- **Hands Reach And Look At:** Character head up vector; look-at target; hand targets.
- **Hold:** Proximity vector; constraint targets; nearest point on surface.
- **Land And Crouch:** Ground height visualisation.
- **Legs Reach:** Leg targets.
- **Stagger:** Ground height visualisation.
- **Tackle:** Ground height visualisation.

Variables

Each behaviour has a set of **behaviour variables**. These are parameters that allow you to control specific aspects of each behaviour individually for each behaviour event. Most behaviour variables are dimensionless parameters in the range 0.0 to 1.0.

The actual motion generated by a behaviour depends not only on its behaviour variables, but also on the dynamics of the character itself, such as its position, orientation and velocity, over the duration of the behaviour event.

Target objects

Specifies the **set of mass objects** to which the behaviour event applies. Mass objects can only be selected from the target character. You cannot edit the mass object names directly. Instead, click the **[Select]** command hotlink and edit the target object set in **Selection** mode.

Keep in mind that behaviour events actually affect **joints**. Although the target object set is composed of mass objects, it is the corresponding parent joint of each target mass object that is modified by the behaviour event.

Click the **[Clear]** command hotlink to clear the set of target mass objects. If the **Target Objects** set is empty, this indicates that the **entire** character is affected by the behaviour event. In contrast to other event types, when the target object set is empty, behaviour events affect the **entire** character. For other event types, if their **Target Objects** or **Target Joints** lists are empty, this would indicate that these event do not affect any object.

Chapter 12

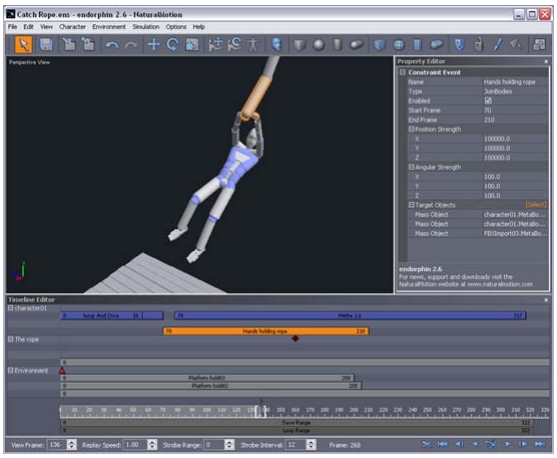
Constraint Events

What are constraint events?

Constraint events allow you to **limit** the motion of mass objects over a range of frames. You can limit the movement or rotation of mass objects, and connect mass objects together so that they share the same movement or rotation. Constraint events are useful when you want to model effects such as holding and catching objects.

You can set the type of constraint event, and specify the strength of the constraint and which mass objects are affected. Constraint events can be applied to any character. The **target object set** for a constraint event can include any number of mass objects from any number of characters.

Constraint events are drawn using a grey **rectangle marker** in the timeline.



Using constraint events

Constraint events are useful in a wide range of circumstances.

Depending on the constraint **type**, and the **strength** of the movement and rotation constraints in each direction, you can obtain different effects.

A common use of constraints is as a way to temporarily **join** objects together into compound objects. You can then model the breakup or shattering of the compound objects according to the scene requirements. For example, if you wanted to model the collision of a character with a piece of furniture, you could use a constraint to hold the furniture together until the moment of impact.

Another use of constraints is to model **catching** or **holding** behaviour. For example, if you wanted to model the effect of a character catching a ball, you could add a constraint between the mass objects associated with the character and the ball, that would be timed to coincide with the impact between the character and ball.

Constraints are often used with mass objects in the environment character. For example, if you want to model the movement of a mass object on a **rail**, you might add a constraint to restrict its motion to a single direction.

Using soft constraints

Hard and soft constraints

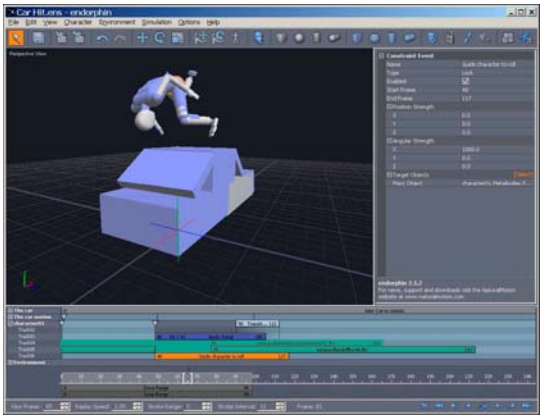
The relative influence that a constraint has during a simulation depends upon its dimensionless positional and angular strength values. The greater the strength value, the more tightly the constraint will influence the motion of its target mass objects.

- When constraint strengths are large—typically 10000.0 or above—constraints are **hard constraints**. This is the default strength for new constraint events. Use hard constraints when you want to rigidly lock mass objects together, or to hold objects into position.
- When constraint strengths are small—typically 100.0 or less—the constraints are **soft constraints**. Use soft constraints to help to guide the motion of mass objects.

Using soft constraints to modify a simulation

Suppose your scene includes a character that is spinning through the air, and a requirement is that the character must land with a particular orientation, such as front-facing.

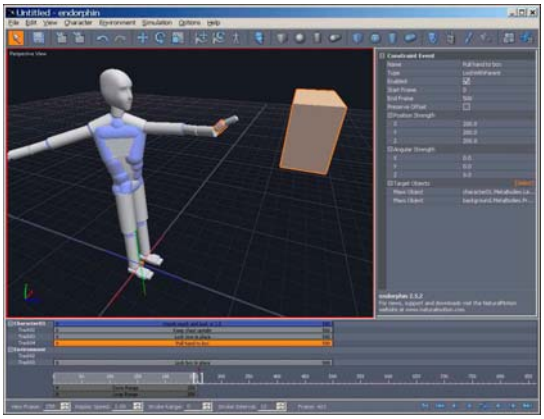
One way of guiding the simulation to achieve this result would be to apply a soft **Lock** constraint to the **root joint** of the character, and specify **zero** positional constraints but **soft** rotational constraints values. The soft rotational constraint will slightly slow down the tumbling motion of the character. You can then tune the rotational constraint strengths to until the desired end orientation is achieved.



Using soft constraints with animated prop characters

Prop characters are often used in *endorphin* scenes to represent vehicles and other moving objects. Typically, these prop characters will be added to a scene, and then animated via animation events or keyframes.

By setting the prop character simulation mode to **Collision** or **Collision With Momentum**, collisions between your animated prop characters and simulation characters will result in momentum transfer from the prop characters to the simulation characters. However, these collisions will **not** affect the motion of the animated prop characters—since they are driven entirely by the keyframes in their animation events or on their timelines. Ideally, the animated prop characters themselves would react to collisions by slightly altering their trajectories.



To add a soft constraint to an animated prop character:

The following steps how you can use soft constraints to mimic the effect of two-way collisions between animated prop characters and other simulated scene objects. The sample scene **HeliLeap.ens** uses this technique. The key idea is to import a prop character and its animation separately, and then using a **soft constraint** to constrain the prop character to its animated trajectory.

1. Build the prop character in **Character Edit Mode**. If you are using Maya, you can create the prop character in Maya and then use the **Maya Exporter MEL script** to create the corresponding *endorphin* .**nmc** prop character.
2. Add this prop character to your scene using **Character > Add Character**.
3. Import the animated motion for the character using **File > Import**. Ensure that you do **not** have the prop character selected. It is important that the motion is imported as a separate character. When importing the character motion, select the appropriate root node for the character in the **Import Options** dialog, and turn on the **Import selected node only** setting.
4. Create a new **constraint event**, and hold down the **Ctrl** key to select the root mass objects of the prop character and the imported motion character.
5. Set the constraint type to **JoinWithParent**.
6. Simulate the scene. Any collisions between the animated prop character and other scene objects will now produce an reaction in the motion of the animated prop character. By adjusting the constraint's positional and angular strengths, you can increase or decrease the degree to which the animated prop responds to collisions.

   When the constraint is extremely **hard**, the animated prop will show little response to collisions. When the constraint is **soft**, the animated prop will show a greater response. Keep in mind that regardless of how weak you make the constraint, the animated prop will eventually return to its animated position.

Using soft constraints to guide object motion

The **LockWithParent** constraint type includes a **Preserve Offset** property that is not available with other constraint types.

- When **Preserve Offset** turned on, child mass objects maintain their relative offsets with respect to the parent mass objects.
- When **Preserve Offset** is turned off, child mass objects will be pulled in towards the parent mass object.

  The strength of the force that pulls the child objects in towards the parent object depends on the positional and angular strengths of the constraint. A useful technique is to use the Preserve Offset feature with relatively low positional and angular constraint strengths in order to gently guide one mass object towards another mass object. Ideally you should use strengths of less than 1000.0 to achieve this effect.

Using preserve offset to guide objects

For example, suppose your scene includes a simulation character whose foot is approaching the ground at a particular frame, and you would like to increase the rate of approach so that the foot hits the ground slightly earlier.

One approach would be to add a small **helper mass object** to the environment at a position that coincides with the intended position of the foot-ground contact, and then a **Lock constraint** to lock the helper mass object in place. You would then add a **soft LockWithParent constraint** with **Preserve Offset** turned off, and hold down the Ctrl key to select the helper mass object and the foot mass object of the simulation character. You can then adjust the strength and timing of the LockWithParent constraint to achieve the desired effect.

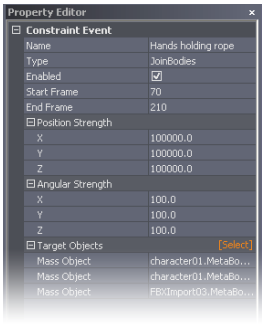
Keep in mind that when you select multiple target mass objects for a LockWithParent constraint, the **last-selected mass object** is interpreted as the parent.

Using preserve offset with behaviours

You can also use this technique in conjunction with the **Hands Reach And Look At** behaviour to extend the reach of a character. You would yse the behaviour to direct the hands of the character to smoothly reach out towards an object, and then apply a soft LockWithParent constraint to help guide the character's hand in towards the target mass object. By using strength values of around 100.0 to 200.0, you can generate convincing shoulder and body twist.

Constraint event properties

Force events have **Name** and **Enabled** properties, which are common to all event types, and **Start Frame** and **End Frame** properties, which are common to all multi-frame event types. In addition, constraint events also have the following properties:



Type

Range: Lock/Lock Position/Lock Orientation/Lock With Parent/Join Bodies. **Default value:** Lock.

Specifies the nature of the constraint.

- **Lock** will prevent the target mass objects from moving or rotating for the duration of the constraint event. This is the default constraint type.
- **Lock Position** will prevent the target mass objects from moving the duration of the constraint event, but allows the objects to rotate.
- **Lock Orientation** will prevent the target mass objects from rotating the duration of the constraint event, but allows the objects to move.
- **Lock With Parent** forces the movement and rotation of the target mass objects to follow the movement and rotation of one particular mass object, which is called the **parent object**, for the duration of the constraint event. By convention, the parent object is the **last** object in the set of target mass objects.

  Use a Lock With Parent constraint type when one mass object is directly affecting other mass objects in a **one-way** direction. This constraint type can be used when one mass object is being driven by animation, and this object is then driving or colliding with mass objects that are being simulated.

  An example might be where a simulation character is sitting on an animated character representing an animal, such as a horse. A Lock With Parent constraint could be used to constrain mass objects in the character with a mass object of the horse character. This would allow the character to sit on the horse and follow the motion of any horse run cycle. Reducing the strength of the constraint would allow some degree of **bounce** for added realism.
- **Join Bodies** forces the movement and rotation of the target mass objects to be locked together. That is, the mass objects can collectively move and rotate as a single rigid body. They cannot move or rotate with respect to each other. This was the default constraint type in some older versions of *endorphin*.

Use a Join Bodies constraint type when you want to model a connection between two or more mass objects in which each object can affect the other objects. This is most often used when all the mass objects are being driven by the dynamic simulation. Join Bodies allows the momentum and linear momentum of each object to affect the other objects in a **two-way** direction.

Position strength

Range: Unlimited positive value. **Default value:** 100000.0.

Specifies the strength of the **movement constraint** as components defined in the global coordinate system.

Does not apply to the Lock Orientation constraint event type. Strengths are dimensionless values that are converted into constraint forces internally. Use very large constraint strengths (in the range 1000000.0) to **rigidly** constrain movement in a given direction. Use small constraint strengths (in the range 1000.0) to provide a **soft** movement constraint in a given direction. Use zero constraint strengths to allow **free** movement in a given direction.

Angular strength

Range: Unlimited positive value. **Default value:** 100000.0.

Specifies the strength of the **rotational constraint** as components defined in the global coordinate system.

Does not apply to the Lock Position constraint event type. Strengths are dimensionless values that are converted into constraint forces internally. Use very large constraint strengths (in the range 1000000.0) to **rigidly** constrain rotation about a given direction. Use small constraint strengths (in the range 1000.0) to provide a **soft** rotation constraint about a given direction. Use zero constraint strengths to allow **free** rotation in a given direction.

Preserve offset

Range: On/Off. **Default value:** On.

Specifies whether or not the child mass objects should maintain their relative offsets with respect to the parent mass objects.

Only available with the **Lock With Parent** constraint type. If this setting is turned off, the child mass objects attempt to move towards the parent mass object. Preserve Offset is turned on by default.

Target objects

Range: Set of mass objects. **Default value:** PelvisMass.

Specifies the set of mass objects to which the constraint applies. At least one mass object must be in this set. The default mass object is the PelvisMass mass object of the target

character. Mass objects can be selected from any character in the scene. You cannot edit the mass object names directly. Instead, click the **[Select]** command hotlink and edit the target object set in **Selection** mode.

If the constraint type is **Join Bodies**, the target mass objects are all rigidly joined to each other. If the constraint type is **Lock With Parent**, the target mass objects are all joined rigidly to the **last** mass object.

Chapter 13

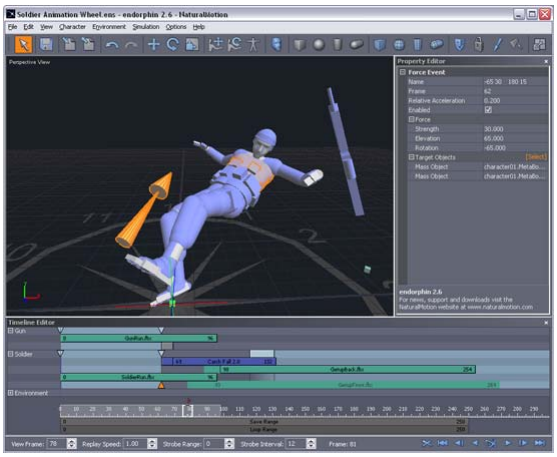
Force Events

What are force events?

Force events allow you to apply a **force** to a set of mass objects at a particular frame. Force events are useful when you want to model effects such as recoil due to collision with a projectile. Force events can also be useful to help guide a dynamic simulation in more subtle ways.

You can set the direction and strength of a force event, and specify which mass objects are affected. Force events can be applied to any character. The **target object set** for a force event can include any number of mass objects from any number of characters.

Force events are drawn using a dark red **triangle marker** in the timeline.



Using force events

Force events are useful when you want to model effects such as recoil due to collision with a projectile. In this case, you might use a single large force event. Force events can also be useful to help influence the trajectories of moving objects. In this case, you might use a **series** of smaller **helper** forces.

Force events are applied to a set of one or more mass objects. You can apply the same force to mass objects belonging to different characters. This means that it does not matter which character timeline track you use to add a force event marker—you can apply the force to mass objects of any character. However, it is good practice to add force event markers to the character track corresponding to the character—or one of the characters—that you are applying the force to.

If you apply the force to **multiple** mass objects, the force is applied **separately** to each mass object. This effectively multiplies the strength of the force by the number of mass objects you are applying the force to.

Using forces over multiple frames

Force events apply an impulse to their target objects at a single frame

However, there are times when you may want to gradually push an object over a number of frames. In the current version of *endorphin*, it is not possible to extend force events to act over multiple frames. This feature may appear in future releases.

The simplest way to mimic the effect of multiframe forces is to create a **series** of force events acting on consecutive frames. Each force event should have its Strength property reduced accordingly.

For example, suppose you want to extend a force event from a single frame into three frames. If the original force event had a strength of 6.0, you should replace it with three force events that each have a strength of 2.0, and ensure that the middle force event is placed at the frame that was originally occupied by the single-frame event. The resulting motion generated by the force events should be similar, but with a more gradual change in direction.

Keep in mind that you can copy-and-paste force events. Use **Edit > Cut**, **Edit > Copy** and **Edit > Paste** to cut, copy and paste selected force events. The keyboard shortcuts are **Ctrl+X**, **Ctrl+C** and **Ctrl+V** respectively.

Single force with strength 6.0. Delivers a sharp impulse and creates an immediate change in direction.

Three forces on consecutive with strength 2.0. Delivers a smoother push with a more subtle change in direction.

Creating forces with offsets

When you apply a force to a single mass object, it is applied at the **centre of mass** of the mass object. Similarly, if you apply a force to multiple mass objects, the force is applied separately to each mass object, at the centre of mass of each mass object respectively.

Sometimes, it is useful to apply a force **away** from the centre of mass of a mass object. This effectively applies a **torque** to the mass object, causing it to rotate in addition to moving. You cannot do this directly in *endorphin*. However, you can use the following technique to achieve the same effect.

To apply a force at an offset to the centre of mass:

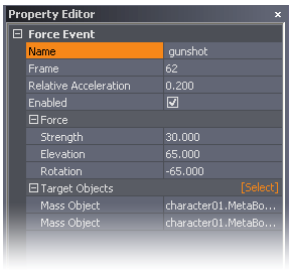
1. In your scene, create a new spherical mass object in the Environment by selecting **Environment > Create Sphere Mass Object**. This will act as a **helper** mass object.
2. Use the Move tool to position the helper mass object at the position that you would like to apply the force.
3. Add a constraint to the Environment by right-clicking on the Environment character timeline and selecting **Create Constraint Event**.
4. With the new constraint selected, hold down the **Ctrl** key and select the helper mass object, as well as the mass object to which you want to apply the force. This adds these mass objects the constraint's **Target Objects** list.
5. With the new constraint selected, set its type to **JoinBodies**.
6. Use the Scale tool to resize the helper mass object so that it as small as possible. It is important that the helper mass object is as small as possible, so that it does not affect the dynamics of the simulation.

7. Apply the force event to the helper mass object. The constraint will transmit the force from the helper mass object to your desired mass object.
8. If you want to increase the relative torque applied by the force, increase the **density** of the helper mass object.

In some cases it may be preferable to create **multiple** helper mass objects, and constrain them to each other and to the target mass object using the JoinBodies constraint. In this case, however, you must still ensure that one of the helper mass objects is positioned at the correct offset. This is the helper mass object to which you must apply the force.

Force event properties

Force events have **Name** and **Enabled** properties, which are common to all event types, and a **Frame** property, which is common to all single-frame event types. In addition, force events also have the following properties:



Strength

Range: 0.01 to 50.0. **Default value:** 12.0.

Specifies the **magnitude** of the force.

This is a dimensionless value that is internally converted into a corresponding physical force.

Elevation

Range: 0.0 to 180.0. **Default value:** 90.0.

Specifies the **vertical angle** of the force vector, in degrees.

A value of 0.0 indicates a force directed vertically upwards. A value of 90.0 indicates a force directed in the horizontal plane. A value of 180.0 indicates a force directed vertically downwards.

Rotation

Range: -180.0 to +180.0. **Default value:** -90.0.

Specifies the **horizontal angle** of the force vector, in degrees.

A value of +90.0 indicates a force directed in the Z direction. A value of -90.0 indicates a force directed in the negative Z direction.

Relative acceleration

Range: -1.0 to +1.0. **Default value:** +0.2.

Specifies the **rate** at which the force is applied.

A value of 0.0 implies a uniform rate of application. Positive values imply a gradual onset of the force. Negative values imply a more rapid application of the force. The Relative Acceleration property has a more noticeable effect when there are multiple target mass objects.

Target objects

Range: Set of mass objects. **Default value:** UpperSpineMass.

Specifies the set of mass objects to which the force applies. At least one mass object must be in this set. Mass objects can be selected from any character in the scene. You cannot edit the mass object names directly. Instead, click the **[Select]** command hotlink and edit the target object set in **Selection** mode.

Force event manipulators

Force events are the only event type that have a corresponding viewport **manipulator**.

To display force manipulators:

- When you create a force event, a corresponding **arrow manipulator** is displayed in the viewports.
- The force manipulator is usually displayed partially **transparently**. However, if the time slider is within three frames of the frame in which the force event is triggered, the force manipulator is displayed **solidly**. In addition, if the time slider is on the frame in which the force event is triggered, the target mass object set is displayed in the viewport using the dark red dependent selection colour.
- Force manipulators are positioned so that the tip of the arrow manipulator coincides with the centre of mass of the target mass object set. If you edit the target mass object set of a force event, the position of the corresponding force manipulator is updated.

To select force manipulators:

- You can **click** on a force manipulator in a viewport to select the corresponding force event.

  Hold down the **Ctrl** key and click on force manipulators to add and remove the corresponding force events from the selection set. You can also use **box selection** to select force manipulators.

To change force rotation and elevation using the Rotate tool

- You can use the **Rotate** tool to rotate force manipulators.

  Rotating a force manipulator effectively changes the **Rotation** and **Elevation** properties of the corresponding force event.

To change force strength using the Rotate tool

- You can use the **Rotate** tool to change the strength of a force.

  When a force manipulator is selected with the **Rotate** tool active, a pink conical **strength manipulator** is displayed at the **rear** of the force manipulator. When you are dragging the strength manipulator to modify the strength of the force, it is displayed using the yellow **highlight colour**.

  Click the strength manipulator and drag it **toward** the force manipulator to reduce the strength of the force.

Click the strength manipulator and drag it **away** from the force manipulator to increase the strength of the force.

To change force strength using the Scale tool

- You can use the **Scale** tool to change the strength of a force.

  When a force manipulator is selected with the **Scale** tool active, you can use the Scale tool manipulator to scale the strength of the force.

  Drag the Scale tool manipulator **toward** from the tip of the arrow manipulator to decrease the strength of the force.

  Drag the Scale tool manipulator **away** from the tip of the arrow manipulator to increase the strength of the force.

Note You can use the **Move** tool to move the move the force manipulator to a different location in the virtual world. However, this is only a visual relocation of the manipulator. It will not affect the simulation in any way. If you make any subsequent changes to the target mass object set, the force arrow manipulator position is reset so that the tip of the force arrow is coincident with the centre of mass of the target mass objects.

Chapter 14

Motion Transfer Events

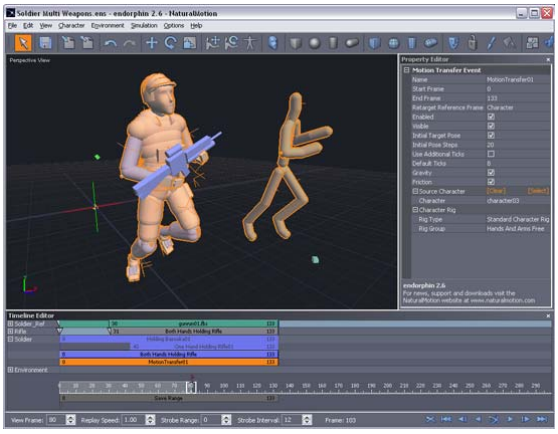
What are *endorphin* motion transfer events?

Motion transfer events allow you to dynamically **map** motion from a reference character to a simulation character.

Motion transfer events are useful when you want to blend animation with other sources of motion, such as behaviours, active poses, forces and constraints. You can specify the **strength** of the motion transfer, and specify which entities in the target character are driven by the motion transfer event.

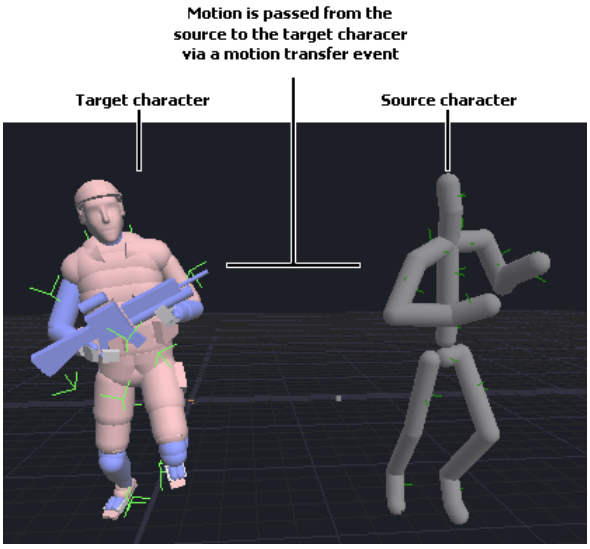
To use a motion transfer event, you need to populate the scene with two characters that form a simulation-reference character pair. The reference character must have been **rigged** to the Standard Character Rig. That is, its mass objects must have been associated with corresponding rig nodes in the Standard Character Rig.

Motion transfer events are used to map motion from a reference character to a simulation character. These are also called the **source** and **target** characters. In most cases, the source reference character will be driven by a passive **animation event**, and the source character will be in either the No Simulation, Collision Only or Collision With Momentum simulation modes. The target simulation character must always be in Full Simulation mode.



Using motion transfer events

To use dynamic motion transfer, you need to add a motion transfer event to the target character timeline and specify the source character.



Specifying the target character

To use dynamic motion transfer, you need to add a motion transfer event to the **target** character that will be receiving the motion. You can add motion transfer events to any character other than the Environment character. However, you will nearly always add motion transfer events to **simulation** characters.

Specifying the source character

The Source Character property defines the source of the motion that is to be transferred to the target character. Initially, no source character is specified, and is indicated by the term **Unspecified Source** appearing as the source character.

Click the **[Select]** hotlink next to the source character property, then select another character to act as the source character. If you make a mistake, click the **[Clear]** hotlink and try again. The source character must be a reference character which forms a simulation-reference character pair with the target simulation character.

For example, suppose you create a **Goblin.nmc** simulation character, and a **Goblin_ref.nmc** reference character. To map motion from **Goblin_ref** to **Goblin**, you will need to add both characters to the scene, then create a motion transfer event on the timeline of **Goblin**, and then select **Goblin_ref** as the source character.

Animating the source character

The source reference character will usually be driven by a passive **animation event**. You will need to select the reference character, and import animation directly onto it, in the form of an animation event.

You will nearly always want to add a simulation marker to the source reference character to set the simulation mode to No Simulation. This will ensure that there are no collisions between the reference character and any other character in the scene. Effectively, setting the simulation mode to No Simulation removes the source character from the scene, **except** for the influence it has via the motion transfer event.

Displaying motion transfer events

When motion transfer events are triggered, the target mass subjects that are being driven are highlighted in the viewports. In addition, the source and target character rig nodes may also be displayed.

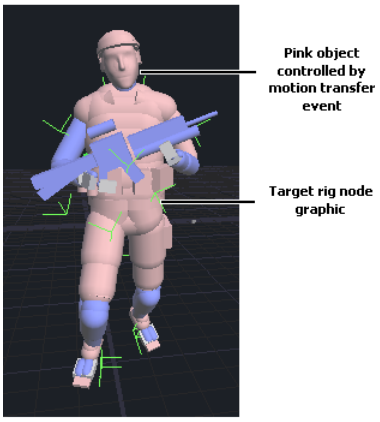
Mass objects

During a simulation, if a motion transfer event is triggered, the target object mass objects that are associated with rig nodes are displayed using a **light red highlight colour** for the duration of the event. Currently, you cannot change this colour.

Rig nodes

During a simulation, if a motion transfer event is triggered—and its **Visible** property is turned on—rig nodes for the source character and target character are displayed in the viewports for the duration of the event. These rig nodes also appear when playing back the animation, and when manually scrubbing through the animation. Rig nodes are displayed at the centre of their corresponding mass objects.

By default, source rig nodes are dark green, and target rig nodes are light green. Their colours can be adjusted in the Display Options editor, using the **Character: Rig Node: Source** and **Character: Rig Node: Target** colour settings respectively.



Changing motion transfer strengths

Motion transfer events include a **Rig Group** property, which specifies which rig group of the target character to use in conjunction with the motion transfer, and a **Rig Type** property, which specifies which character rig type to use.

Rig groups

Dynamic motion transfer involves driving mass objects on a target character using the motion of corresponding mass objects on a source character. The source and target mass objects each are associated with a shared rig node, whereby the rig node **strength** specifies the degree to which the rig node drives the corresponding target mass object.

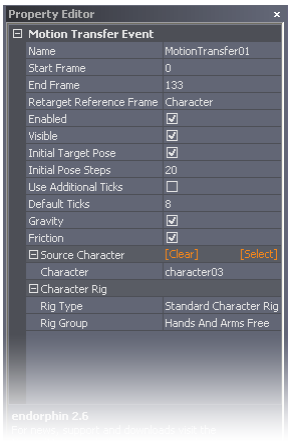
Each character contains one or more **rig groups**, which are sets of rig node strengths. All characters contain the **Standard Motion Transfer** rig group, but you can create additional rig groups in Character Edit Mode. For example, you may define a rig group in which rig node strengths are only non-zero for the arm mass objects. A motion transfer event that uses this rig group would only drive the arms of the target character.

Rig types

You can also choose which character **rig type** to use for the motion transfer by selecting from the list of rigs in the Rig Type property. However, in *endorphin* 2.5, the only rigs available are the Standard Character Rig and the Advanced Character Rig. You will nearly always use the Standard Character Rig. The Advanced Character Rig will be used in later releases of *endorphin*.

Motion transfer event properties

Motion transfer events have **Name** and **Enabled** properties, which are common to all event types, and **Start Frame** and **End Frame** properties, which are common to all multi-frame event types. In addition, motion transfer events also have the following properties:



Retarget reference frame

Range: Character/World. **Default value:** Character.

Specifies whether to map the motion in terms of **target character space** or **world space**.

When this property is set to target character space, the relative separation and relative orientation of the source and target characters is preserved. When this property is set to world space, the target character is superimposed over the source character during the motion transfer event. Most of the time you will use target character space.

Visible

Range: On/Off. **Default value:** On.

Specifies whether to display rig nodes during a simulation.

When this setting is on, the rig nodes associated with each mass object in the source character and target character are displayed in the viewport. Source character rig nodes are displayed using the dark green source rig node colour. Target character rig nodes are displayed using the light green target rig node colour. Note that even when this setting is turned on, rig nodes are still only displayed for frames corresponding to the duration of the motion transfer event.

Initial target pose

Range: On/Off. **Default value:** On.

Specifies whether to ensure that the root node of the target character is aligned to the root node of the source character at the start frame. You will usually keep this setting turned on unless you want to blend into motion of the target character more gradually.

Initial pose steps

Range: Unlimited positive value. **Default value:** 20.

Specifies how many simulation steps to use to ensure that the initial pose of the target character settles into the correct pose as specified by the source character. Increasing the number of steps produces a more accurate result.

Use additional ticks

Range: On/Off. **Default value:** Off.

Specifies whether to use additional simulation steps when performing dynamic motion transfer. Use this setting when you need more accurate motion transfer.

Default ticks

Range: Unlimited positive value. **Default value:** 8.

Specifies how many additional simulation steps to use when the **Use additional ticks** setting is turned on. Using more simulation steps can improve the accuracy of motion transfer. Keep in mind that this setting will also slow the overall simulation rate, so do not use a higher value than necessary.

Gravity

Range: On/Off. **Default value:** On.

Specifies whether to consider gravity during the dynamic motion transfer event. Overrides the default simulation setting.

Friction

Range: On/Off. **Default value:** On.

Specifies whether to consider friction during the motion transfer event.

Source character

Range: Reference character. **Default value:** [Unspecified source].

Specifies the character that is acting as the motion source.

Click the **[Select]** command hotlink to enter **Selection** mode, and then select a source character in the viewport or in the Timeline Editor. Click the **[Clear]** command hotlink to clear the source character. The source character should always be the reference character that corresponds to the target simulation character.

Rig type

Range: Character rig. **Default value:** Standard Character Rig.

Specifies which target character rig to use during dynamic motion transfer. In *endorphin* 2.5, this should always be set to Standard Character Rig.

Rig group

Range: Character rig group. **Default value:** Standard Motion Transfer.

Specifies which target character rig group to use during dynamic motion transfer.

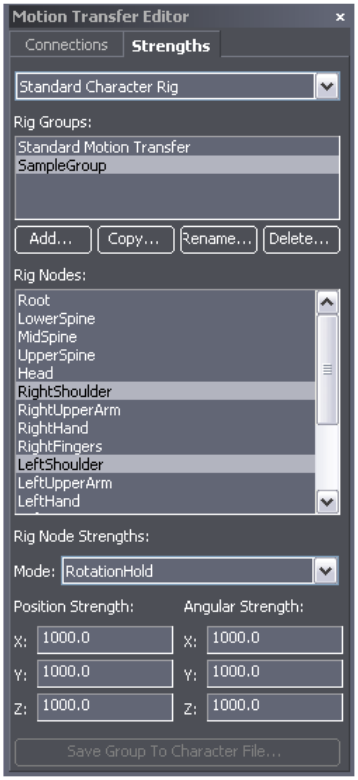
By default, this will be set to the **Standard Motion Transfer** rig group, which is common to all simulation characters. The Standard Motion Transfer group transfers all positional and angular motion between all mass objects.

If the target simulation character has additional rig groups defined, they will appear in the Rig Group list. You can use different rig groups to achieve different motion transfer effects.

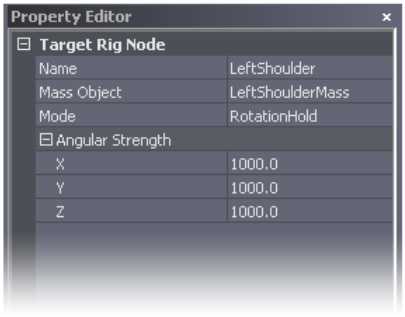
Editing target rig node strengths

When you are creating and editing characters, you can use the **Motion Transfer Editor** in Character Edit Mode to create and edit **rig groups** for that character. Each rig group is a different configuration of rig node strengths. For example, you might create a rig group in which only rig nodes associated with arm mass objects have non-zero strength.

Rig node strength in motion transfer editor



Rig node strength in Property editor



Editing rig node strengths in scenes

You can also use the Motion Transfer Editor to create and edit rig groups for characters within **scenes**. The distinction between editing characters in Character Edit Mode and editing characters in scenes is that when you edit a character in a scene, you are editing an **instance** of that character, rather than the definition of the character.

One reason why you might edit a character in the context of a scene is that it is often useful to **fine-tune** rig node strengths for a character by running various simulations in which the character is driven by a motion transfer event. After each simulation, you can modify the target character rig node strengths in order to achieve a desired effect.

There are two ways to edit target rig node strengths. One way is to use the Motion Transfer Editor. The other way, which is often more convenient, is to **select** the desired target rig node directly in the viewport. When a target rig node is selected in a viewport, the Property Editor displays its corresponding properties, allowing you to modify its hold mode and hold strengths directly. In order to select target rig nodes, you must ensure that the **Visible** property of the motion transfer event is turned on.

Note You cannot change rig node modes or strengths when the motion transfer event is using the **Standard Motion Transfer** rig group. This rig group is common to all characters and cannot be edited.

Copying rig node strengths to the character definition

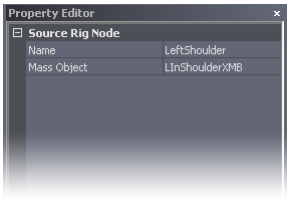
After you have fine-tuned the rig node strengths of an **instance** of a character in a scene, you may want to copy these settings into the character definition **.nmc** file. This is necessary if you want to reuse these settings in other scenes.

You can do this using the Motion Transfer Editor:

1. Select **View > Motion Transfer Editor** to display the Motion Transfer Editor. The keyboard shortcut is **M**.
2. Select a character using the viewport, Timeline Editor or Node View. The character selection is **sticky**. This means that once you have selected a character, you can deselect it without losing its data in the Motion Transfer Editor.
3. Select the rig group from the **Rig Groups** listbox.
4. Click the **Save Group To Character File…** button to copy this rig group settings into the character definition file.

Source rig node properties

Source rig nodes are associated with mass objects of the character used as the motion **source** during a motion transfer event. They are only displayed when the Visible setting of the motion transfer event is turned on. None of the properties of a motion transfer event can be edited.



Name

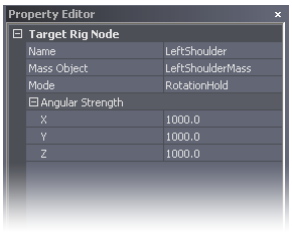
Specifies the source rig node. Cannot be edited.

Mass object

Specifies which mass object is associated with the source rig node. Cannot be edited using the Property Editor. To edit associations between rig nodes and mass objects of the source character, use the **Motion Transfer Editor**.

Target rig node properties

Target rig nodes are associated with mass objects of the character used as the motion **target** during a motion transfer event. They are only displayed when the Visible setting of the motion transfer event is turned on.



Name

Specifies the target rig node. Cannot be edited.

Mass object

Specifies which mass object is associated with the target rig node. Cannot be edited using the Property Editor. To edit associations between rig nodes and mass objects of the target character, use the **Motion Transfer Editor**.

Mode

Range: FullHold / FullTranslationHold / FullRotationHold / NoHold / TranslationHold / RotationHold / TranslationAndRotationHold. **Default value:** Full Hold

Specifies how the rig node drives the target mass object. You can specify whether the rig node drives positions, or rotations, or both. You can also specify whether the rig node drives the mass object at **full** strength or at a strength that you specify.

- **FullHold** drives positions and rotations of the target mass object.
- **FullTranslationHold** drives positions, but not rotations, of the target mass object.
- **FullRotationHold** drives rotations, but not positions, of the target mass object.
- **NoHold** does not drive positions or rotations of the target mass object. This is equivalent to disabling the rig node.
- **TranslationHold** drives positions, but not rotations, of the target mass object. However, unlike FullTranslationHold mode, TranslationHold mode has an upper limit on its strength that you specify in the position strength property.

- **RotationHold** drives rotations, but not positions, of the target mass object. However, unlike FullRotationHold mode, RotationHold mode has an upper limit on its strength that you specify in the rotation strength property.
- **TranslationAndRotationHold** drives positions and rotations of the target mass object. However, unlike FullHold mode, TranslationAndRotationHold mode has upper limits on its strength that you specify in the position and rotation strength properties.

Position strength

Range: Unlimited positive value. **Default value:** 1000.0.

Specifies the upper limit of the strength applied to drive the target mass object along each of its local X, Y and Z directions. Strength is a dimensionless parameter. Use larger values (around 1000.0) for relatively strong connections. Use smaller values for weaker holds. Use a zero strength value for zero hold. The position strength is only used when the target rig node mode is TranslationHold or TranslationAndRotationHold.

Angular strength

Range: Unlimited positive value. **Default value:** 1000.0.

Specifies the upper limit of the strength applied to drive the target mass around each of its local X, Y and Z directions. Strength is a dimensionless parameter. Use larger values (around 1000.0) for relatively strong connections. Use smaller values for weaker holds. Use a zero strength value for zero hold. The angular strength is only used when the target rig node mode is RotationHold or TranslationAndRotationHold.

Improving motion transfer with active poses

Underconstrained joint motion

When motion is dynamically mapped from source characters to target characters, mass objects on the source character drive mass objects on the target character.

However, only mass objects that have been assigned to rig nodes are involved in dynamic motion transfer. The position and orientation of unassigned target character mass objects may be **underconstrained**. As a result, the motion of these mass objects may be undefined, leading to unwanted random motion in their corresponding joints.

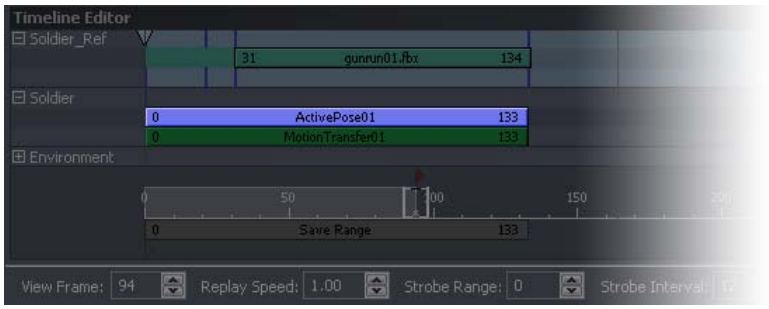
This problem only occurs when you are using dynamic motion transfer in conjunction with **motion transfer events**. It is not an issue when using dynamic motion transfer in the Auto Motion Transfer phase of animation import or export. In the case of animation import and export, the position and orientation of unassigned joints is resolved automatically.

Adding an active pose

The simplest way of dealing with this issue is to apply an **active pose event** to the target character. The start frame of the active pose event should match the start frame of the motion transfer event. The end frame of the active pose event should match the end frame of the animation event that is being mapped by the motion transfer event. The **strength** of the active pose can be adjusted to control the degree to which you want to smooth the motion of the unassigned mass objects.

Most of the time, you will use this technique to smooth the motion of the extremities of the target character, such as its finger and toe joints. As such, this technique will work with most active poses, such as a T-poses. However, if you have specific requirements for the positions of unassigned joints, then you should edit the active pose acoordingly.

The active pose technique allows you to quickly add overall stiffness to a target character. If you require more detailed control on a per-joint basis, you can also adjust the target character joint limits in Character Edit Mode. This is a more complicated method, but does allow for greater control over the motion transfer process.



Chapter 15

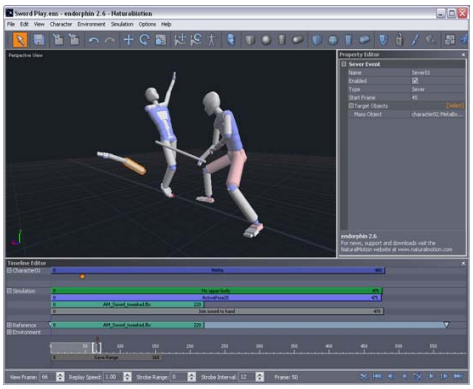
Sever Events

What are sever events?

Sever events allow you to **disconnect** and **reconnect** joints at a particular frame. Sever events are useful when you want to model the separation of articulated structure into separate sub-structures.

You can set the **type** of a sever event, and specify which mass objects are affected. Sever events can be applied to any character. The **target object set** for a sever event can include any number of mass objects from any number of characters. Although you apply a sever event to mass objects, it is actually the corresponding parent joint of each target mass object which is disconnected or reconnected.

Sever events are drawn using a dark red **diamond marker** in the timeline.



Using sever events

There are various **types** of sever event.

The most common use of sever events is when you want to model effects such as the **detaching** of joints. This allows you to break apart articulated structures such as characters during the course of a simulation.

However, you can use sever events in other ways, such as to reattach joints. You can also use sever events to change other settings, such as to enable or disable joint stiffnesses.

When you work with sever events, you select **mass objects** to specify the target object set. However, it is the corresponding parent joints of the target mass objects that are actually modified.

Using sever events with animation export

If you use **sever events** during a simulation to sever one or more joints of a simulation character, you cannot simply export the simulation character motion onto its corresponding reference character using Auto Motion Transfer. Scenes involving sever events applied to simulation characters must be set up in the following way.

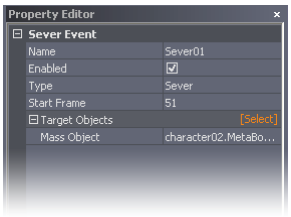
To use sever events with animation export:

1. Add the **simulation** character to a scene. In addition, add its corresponding **reference** character to the same scene.
2. Add simulation events to the first frame of both simulation and reference character timelines. Set the simulation mode of the simulation character to **Full Simulation** and the simulation mode of the reference character to **No Simulation**. This ensures that the reference character is not simulated and plays no part in the scene.
3. Create the required sever events. Ensure that the **Target Objects** set for the sever events include the corresponding joints in **both** the simulation and reference characters. For example, if you apply the sever to the **neck joint** of the simulation character, ensure that you also select the neck joint of the reference character and add it to the sever event selection set.
4. Simulate the scene as required.
5. **Bake** the animation of the simulation character by right-clicking on its timeline and selecting **Create All Keys**.
6. Set the simulation mode of the simulation character to **No Simulation**, and the simulation mode of the reference character to **Full Simulation**. Also, move the reference character to a position in the scene that ensures that it will not collide with any other character in the scene, including the environment.

7. Add a **motion transfer event** to the reference character timeline, and select the simulation character as the source character of the motion transfer.
8. Resimulate. Motion is now transferred from the simulation character to the reference character, and the sever events will sever joints on the reference character at the correct frames.
9. You can now select the reference character and export its motion.

Sever event properties

Sever events have **Name** and **Enabled** properties, which are common to all event types, and a **Frame** property, which is common to all single-frame event types. In addition, force events also have the following properties:



Type

Range: Sever/Reattach/Sever Switch Off Children/Disable/Enable/Disable Children/Enable Children. **Default value:** Sever.

Keep in mind that sever events affect **joints**. Although the target object set is composed of mass objects, it is the corresponding parent joint of each target mass object that is modified by the sever event.

- **Sever** will **disconnect** the selected joints at the specified frame. Each collection of articulated joints, mass objects and collision objects on either side of a disconnected joint becomes a separate body that can move and rotate independently.
- **Reattach** will **reconnect** the selected joints at the specified frame. It is only meaningful to reattach a joint that has been previously severed. Reattaching a joint that has not be previously severed has no effect. When a severed joint is reattached, it rapidly reverts to its original length and position relative to its parent joint.
- **Sever Switch Off Children** will **disconnect** the selected joints at the specified frame. In addition, it will convert any child joints further down the joint hierarchies into **passive** joints. This means that their joint limits will be ignored, and they will no longer be driven by behaviours or active poses.

- **Disable** will **deactivate** joint stiffness for the selected joints at the specified frame.
- **Enable** will **reactivate** joint stiffness for the selected joints at the specified frame.
- **Disable Children** will **deactivate** joint stiffness for the selected joints at the specified frame. In addition, it will deactivate the joint stiffness for any child joints further down the joint hierarchy of each selected joint.
- **Enable Children** will **reactivate** joint stiffness for the selected joints at the specified frame. In addition, it will reactivate the joint stiffness for any child joints further down the joint hierarchy of each selected joint.

Target objects

Range: Set of mass objects. **Default:** LowerSpineMass.

Specifies the **set of mass objects** to which the sever event applies. At least one mass object must be in this set. Mass objects can be selected from any character in the scene. You cannot edit the mass object names directly. Instead, click the **[Select]** command hotlink and edit the target object set in **Selection** mode.

Keep in mind that sever events actually affect **joints**. Although the target object set is composed of mass objects, it is the corresponding parent joint of each target mass object that is modified by the sever event.

Chapter 16

Simulation Events

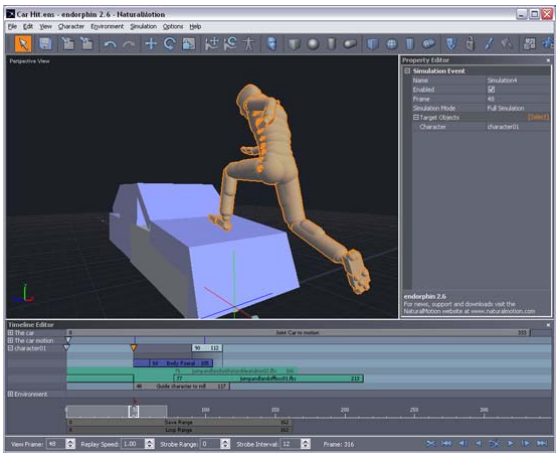
What are simulation events?

Simulation events are used when you want to incorporate keyframed data into a scene.

Simulation events control the **simulation mode**. By changing the simulation mode, you can specify whether a character is being dynamically simulated or fully driven by animation keyframes at any given frame.

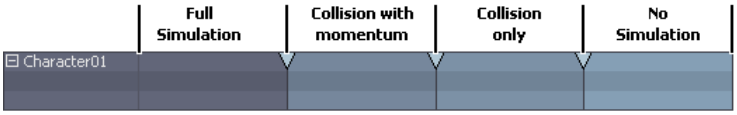
You can add multiple simulation events to a character timeline. This is useful if you want to connect many elements of animation data with *endorphin* dynamic simulations.

Simulation events are drawn using a light blue grey **triangle marker** in the timeline. Unlike most other event markers, simulation event markers are always locked to the topmost track in the character timeline. (Transition events are also locked to the topmost track.)



Introducing simulation modes

Simulation events control the **simulation mode** for each character and for the Environment.



Dynamic simulation

When characters are **dynamically simulated**, their motion is dynamically generated as part of the virtual world simulation. They are affected by physical conditions, such as gravity and friction, and physical processes, such as collisions with other characters. They are also influenced by timeline events such as behaviours, active poses, forces, constraints, severs and **Active** animation events.

The corresponding simulation mode for dynamic simulation is the **Full Simulation** mode. By default, all characters—including the environment character—are in Full Simulation mode.

Keyframe animation

When characters are fully driven by **keyframe animation**, their motion is based upon keyframes contained in animation events or on the character timeline. They are *not* influenced by physical processes, or by timeline events such as forces and behaviours.

The corresponding simulation modes for fully keyframe driven animation are the **No Simulation**, **Collision Only** and **Collision With Momentum** modes. You can add simulation events to change the simulation mode to one of these modes

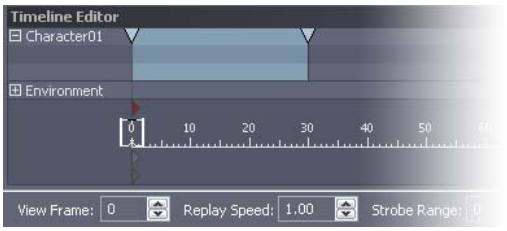
Using simulation events

Transitioning to dynamic simulation

The main use of simulation events is for animation-to-simulation transitions.

When transitioning from any of the keyframe animation simulation modes—No Simulation, Collision Only or Collision With Momentum—to Full Simulation mode, *endorphin* will create a smooth dynamic transition. All positions, orientations and velocities of the character at the corresponding frame of the simulation event will be carried forward into the dynamic simulation.

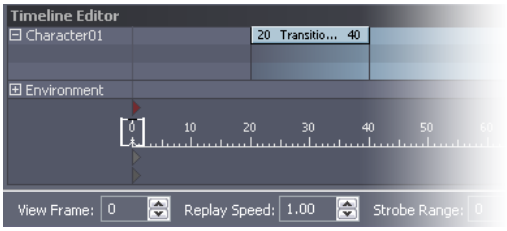
If a simulation event occurs **after** the last available animation key, the dynamic simulation will begin with zero velocity. To avoid this, ensure that the simulation event lies before or coincides with the last animation key.



Transitioning to keyframe animation

You will generally not use simulation events for **simulation-to-animation** transitions.

When transition from Full Simulation mode to any of the fully keyframe driven animation modes—No Simulation, Collision Only or Collision With Momentum—the first frame of the animation will almost certainly not match the last frame of the dynamic simulation. Simply using a simulation event will usually lead to motion discontinuities. Instead, it is better to use **transition events**. Transition events are similar to simulation events, but are specifically designed to ensure smooth transitions from dynamic simulations back to keyframed animation.

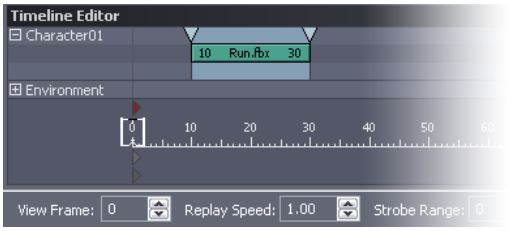


Importing animation

By default, when you **import** animation data onto a character, *endorphin* adds simulation events at the start and end frames of the animation event or animation keyframes. The first simulation event sets the simulation mode to No Simulation, and the second simulation event sets the mode to Full Simulation. These events allow the character to be fully driven by the animation over the duration of the animation event.

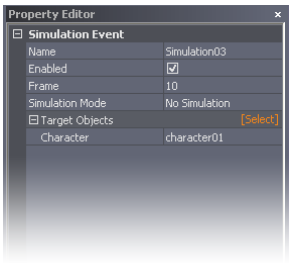
You can edit these events to change their timing and simulation level. For example, by moving the simulation events closer together, you can effectively clip the range of the animation.

If the animation data is to be used after a period of dynamic simulation, you will generally need to delete the initial simulation event and replace it with a transition event to ensure a smooth blend from the dynamic simulation to the animation.



Simulation event properties

Simulation events have **Name** and **Enabled** properties, which are common to all event types, and a **Frame** property, which is common to all single-frame event types. In addition, simulation events also have the following properties:



Simulation mode

Range: Full Simulation/Collision With Momentum/Collision Only/No Simulation. **Default value:** Full Simulation.

Specifies whether a character has motion generated by the dynamic simulation, or whether it is driven fully by keyframe animation. If driven by animation, the mode specifies how the character interacts with other characters and props in the scene.

- In **No Simulation** mode, the character does not collide with any other characters in scene.

  You will commonly use this mode when you have added both a simulation character and its corresponding reference character to a scene, and you are using motion transfer events to map motion from the reference character to the simulation character. By setting the reference character to use No Simulation mode, you ensure that it is does not influence the scene in a direct way. It could still influence it via a motion transfer event, or by a constraint event, which may uses some part of the character's position and motion as a source for the constrained object.

- In **Collision Only** mode, the character can collide with other characters in the scene, provided that the other characters are in Full Simulation mode.

Collisions will not affect the motion of the Collision Only character, but will affect the motion of any Full Simulation characters that it collides with. No momentum is transferred.

You will commonly use this mode with prop characters that do not need to move or only move slowly during a simulation. By setting a prop character to Collision Only mode, you ensure that other simulated characters can collide with the prop, which modifies their motion without affecting the prop.

- In **Collision With Momentum** mode, the character can collide with other characters in the scene, provided that the other characters are in Full Simulation mode, and impart momentum to those characters.

  Collisions will not affect the motion of the Collision With Momentum character, but will affect the motion of any Full Simulation characters that it collides with. Momentum is transferred, and friction is taken into account.

  You should use this mode when the character is being driven by animation data and you want him to collide with other characters or props. This mode ensures that collisions are physically realistic, and that momentum is transferred during collisions. This mode is slower to simulate than Collision Only mode, but generates more accurate collisions.

Target objects

Range: Character. **Default value:** Corresponding character.

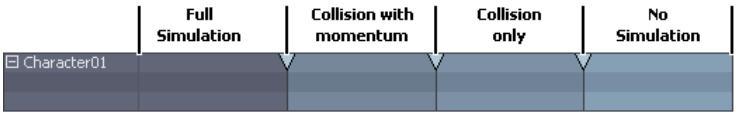
Specifies the character to which the simulation event applies. By default, this is the character corresponding to the timeline track to which the event is applied. Do not change this property.

Simulation mode colours

During the course of a simulation, a given character may be simulated for certain frame ranges, and unsimulated for other frame ranges. You can identify whether or not a character is being simulated in the following ways:

Simulation modes in the Timeline Editor

When a character is in one of the unsimulated modes, a **blue-grey overlay tint** is applied to the corresponding character track in the Timeline Editor.



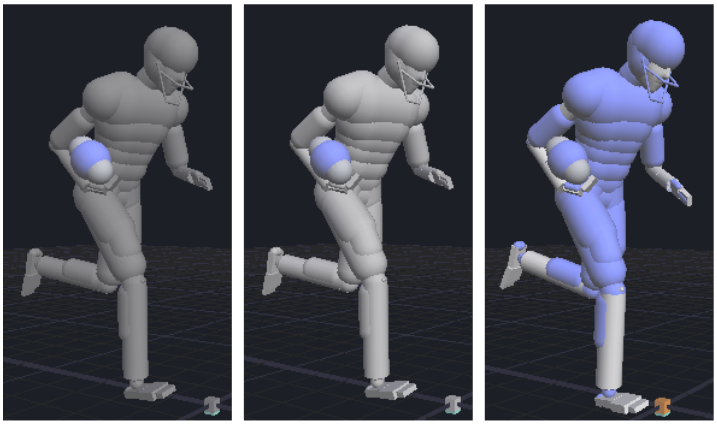
The degree of transparency of the overlay reflects the degree to which the character interacts with other characters in the scene. As the simulation mode is changed from **No Simulation** to **Collision Only** to **Collision With Momentum**, the overlay becomes progressively more transparent.

You can change the colour and transparency of this tint by changing the **Timeline Editor: Simulation Level** system colour.

Simulation modes in the viewports

When a character is in one of the unsimulated modes, the character's mass objects, collision objects and graphical objects are displayed in the viewports using the **grey simulation level colour**.

The specific shade of grey reflects the degree to which the character interacts with other characters in the scene. As the simulation mode is changed from No Simulation to Collision Only to Collision With Momentum, the shade of grey becomes progressively lighter. You can change this colour by changing the **Character: Simulation Level** system colour.



No simulation | Collision only
Collision with momentum | Full Simulation

Chapter 17

Transition Events

What are transition events?

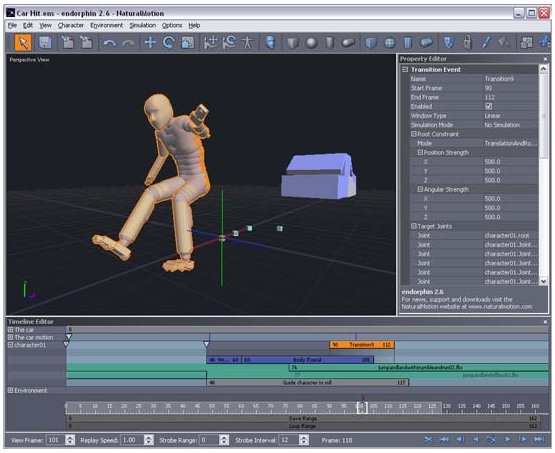
Transition events are used when you blend between different sections of animation. They are a type of **simulation event** that has been designed for blending from dynamic simulation to fully keyframe driven animation

Transition events and simulation events both control the **simulation mode**. By changing the simulation mode, you can specify whether a character is being dynamically simulated or driven by animation keyframes at any given frame. However, transition events work over a period of frames, whereas simulation events occur at a single frame.

You can add multiple transition events to a character timeline. This is useful if you want to connect many elements of animation data with *endorphin* dynamic simulations.

Transition events are drawn using a light blue grey **rectangle marker** in the timeline. Unlike most other event markers and like simulation event markers, transition event markers are always locked to the topmost track in the character timeline.

Transition events can be physically overlapped in time, but the results are undefined.



Transition event simulation modes

Transition events and simulation events control the **simulation mode** for each character. During a simulation, characters will either be dynamically simulated, or will be fully driven by keyframe animation. Transition events are a special type of simulation event designed to blend from dynamic simulation to keyframed animation.

Dynamic simulation

When characters are **dynamically simulated**, their motion is dynamically generated as part of the virtual world simulation. They are affected by physical conditions, such as gravity and friction, and physical processes, such as collisions with other characters. They are also influenced by timeline events such as behaviours, active poses, forces, constraints, active animation and sever events.

The corresponding simulation mode for dynamic simulation is the **Full Simulation** mode. By default, all characters—including the environment character—are in Full Simulation mode.

Keyframe animation

When characters are fully driven by **keyframe animation**, their motion is based upon keyframes contained in animation events or on the character timeline. They are *not* influenced by physical processes, or by timeline events such as forces and behaviours.

The corresponding simulation modes for fully keyframed animation are the **No Simulation**, **Collision Only** and **Collision With Momentum** modes. You can add simulation events to change the simulation mode to one of these modes.

Using transition events

Transition events are a special type of simulation event designed to **blend** from dynamic simulation to keyframed animation. In *endorphin* 2.6 only **linear** blending is supported. Future releases will expand on this option.

At the start of a transition event, the character is placed in Full Simulation mode. By the end of the transition event, the character will be completely set to the desired keyframed animation mode (No Collision, Collision Only or Collision With Momentum), and the position of the character will exactly match the corresponding animation key. At intermediate frames, the position and orientation of the character will progressively blend from its dynamically generated motion into its fully keyframed motion.

You will typically place a transition event at the **start** of the animation that you want to blend into. For example, suppose you have an animation event between frames 100 and 200, and you are dynamically simulating between frames 0 and 99. You might add a transition event between frames 100 and 120, for example, to help blend smoothly between the dynamic simulation and the animation event.

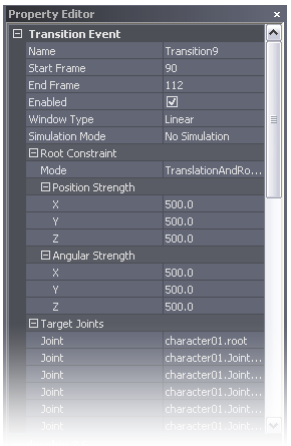
Techniques for better blends

There are a number of techniques to use in conjunction with transition events. These techniques help ensure that your transitions are as natural as possible, and can significantly improve the quality of your overall simulations.

- Transition events are most effective when the position, orientation and motion of the character at the end of the dynamic simulation is not too different from the motion contained in the keyframed animation.
- You can use **active animation events** to help improve transitions from dynamic simulation to keyframed animation. Active animation events are a way of driving dynamically simulated characters using animation keyframes. In the example above, you might extend the animation event to fit between frames 80 and 200, and specify the animation event to be **active**. Between frames 80 and 99, the character will be in Full Simulation mode, but the active animation event will help influence its motion so that it more closely matches the motion at the start of and over the duration of the event.
- You can use other timeline events—such as **forces**, **constraints** and **motion transfer** events—to help guide the motion of the character while it is in Full Simulation mode so that it more closely matches the animation event that is being blended into. Typically these helper events should be quite subtle, so that the physical realism of the motion is maintained as far as possible.
- When you are using transition events to blend between two animation events, you should ensure that the **first** animation is placed on a **lower** timeline track than the second animation event. This ensures that the second animation event has priority over the first animation event.

Transition event properties

Transition events have **Name** and **Enabled** properties, which are common to all event types, and **Start Frame** and **End Frame** properties, which are common to all multi-frame event types. In addition, constraint events also have the following properties:



Window type

Range: Linear. **Default value:** Linear.

Specifies the blend transition type. Currently only linear blends are available.

Simulation mode

Range: Collision With Momentum/Collision Only/No Simulation. **Default value:** No Simulation.

Specifies the final simulation mode of the character at the end of the transition event.

- In **No Simulation** mode, the character does not collide with any other characters in scene.

  You will commonly use this mode when you do not wish the corresponding character to collide with any objects in the scene.

- In **Collision Only** mode, the character can collide with other characters in the scene, provided that the other characters are in Full Simulation mode.

  Collisions will not affect the motion of the Collision Only character, but will affect the motion of any Full Simulation characters that it collides with. No significant momentum is transferred.

You will commonly use this mode with prop characters that do not need to move—or only move slowly—during a simulation. By setting a prop character to Collision Only mode, you ensure that other simulated characters can collide with the prop, which modifies their motion without affecting the prop.

- In **Collision With Momentum** mode, the character can collide with other characters in the scene, provided that the other characters are in Full Simulation mode, and impart momentum to those characters.

  Collisions will not affect the motion of the Collision With Momentum character, but will affect the motion of any Full Simulation characters that it collides with.

  This mode ensures that collisions are physically realistic, and that momentum is transferred during collisions. This mode is slower to simulate than Collision Only mode, but generates more accurate high velocity collisions.

Root constraint mode

Range: FullHold / FullRotationHold / FullTranslationHold / TranslationAndRotationHold / RotationHold / TranslationHold / NoHold. **Default value:** TranslationAndRotationHold.

Specifies if the character's root node should have an extra constraint applied to it during the transition event. By default, transition events will move and rotate the character root during the transition event so that the character is aligned with the animation keyframes. At the same time, the rotations of each joint are blended with the rotations in the animation keyframes. The root node is treated as a special case because transitions are often more physically plausible when the character's root is constrained to follow that of the target animation.

- **TranslationAndRotationHold** will move and rotate the character root node during the duration of the timeline event using the specified positional and angular strengths. You will nearly always use this mode.
- **TranslationHold** will move the character root node during the duration of the timeline event using the specified positional strengths.
- **RotationHold** will rotate the character root node during the duration of the timeline event using the specified angular strengths.
- **NoHold** will not apply any special movement or rotation to the root node during the duration of the timeline event. Instead, the root node is treated like any other character joint. Typically this will make the transition more gradual.
- **FullHold** will immediately move and rotate the root to match the animation. Rarely used.
- **FullTranslationHold** will immediately move the root to match the animation. Rarely used.
- **FullRotationHold** will immediately rotate the root to match the animation. Rarely used.

Position strength

Range: Unlimited positive value. **Default value:** 500.0.

Specifies the strength of the root node **movement constraint** as components defined in the global coordinate system.

Only applies to the TranslationHold and TranslationAndRotationHold modes. Use larger values to move the root node more rapidly into position, and use smaller values to move the root node less rapidly into position. As a general rule, try to use the smallest value that successfully transitions between the simulation and the animation. If the position strength is too small or too large, the transition may not be smooth enough to be physically realistic.

Angular strength

Range: Unlimited positive value. **Default value:** 500.0.

Specifies the strength of the root node **rotational constraint** as components defined in the global coordinate system.

Only applies to the RotationHold and TranslationAndRotationHold modes. Use larger values to rotate the root node more rapidly into position, and use smaller values to rotate the root node less rapidly into position. As a general rule, try to use the smallest value that successfully transitions between the simulation and the animation. If the angular strength is too small or too large, the transition may not be smooth enough to be physically realistic.

Target joints

Range: Set of joints on the target character. **Default value:** All joints of the target character.

Specifies the set of joints which are blended by the transition event. Currently, all joints are affected, and this set cannot be edited. In future versions of *endorphin*, this set may become editable.

Transition event colours

During the course of a transition event, the corresponding character transitions from dynamic simulation (Full Simulation) to one of the keyframe animation modes (No Simulation, Collision Only or Collision With Momentum). You can identify the degree to which the character is simulated or animated in the following ways:

Transition events in the Timeline Editor

During the period of a transition event, a **blue-grey overlay tint** is applied to the corresponding character track in the Timeline Editor. Initially, the overlay is fully transparent. Over the course of the transition event, the overlay becomes less transparent, to reflect the increasing influence of the animation data.

The final degree of transparency of the overlay reflects the degree to which the character interacts with other characters in the scene. Depending on whether the final simulation mode is No Simulation, Collision Only or Collision With Momentum, the final overlay transparency becomes progressively more transparent.

You can change the colour and transparency of this tint by changing the **Timeline Editor: Simulation Level** system colour.

Transition events in the viewports

During the period of a transition event, a **grey overlay tint** is applied to the corresponding character mass objects, collision objects and graphical objects in the viewports. Initially, the overlay is fully transparent. Over the course of the transition event, the overlay becomes less transparent, to reflect the increasing influence of the animation data.

The final shade or grey of these objects reflects the degree to which the character interacts with other characters in the scene. Depending on whether the final simulation mode is No Simulation, Collision Only or Collision With Momentum, the final shade of grey becomes progressively lighter.

You can change this colour by changing the **Character: Simulation Level** system colour.

Chapter 18

Behaviour Library

Introducing the Behaviour Library

Behaviour events allow you to add high-quality humanlike movement to characters in a dynamic simulation. You can choose the behaviour **type** from the behaviours in the **Behaviour Library**. Behaviours are categorized as follows:

Hand and arm behaviours

There are a number of behaviours that are designed to affect the motion of the **hands** and **arms** of the simulation character. These are: Arms Crossed On Chest; Arms Raised Above Head; Arms Wide Of Head; Arms Windmill; Arms Zombie; Hands Covering Face; Hands From Behind Back; Hands Protecting Groin; and Hands Reach And Look At.

Leg behaviours

There are a number of behaviours that are designed to affect the motion of the **legs** of the simulation character. These are: Legs Kick; Legs Reach; and Legs Straighten.

Whole body behaviours

There are a number of behaviours that are designed to affect the motion of the **entire body** of the simulation character. They include Balance; Balance With Props; Body Foetal; Catch Fall; Fall Back, Twist And Catch Fall; Fall Back, Twist And Cover Face; Fall Back, Twist With Passive Arms; Jump; Jump And Dive; Land And Crouch; Stagger; Tackle; Writhe; and Writhe In Mid-Air.

Other behaviours

There are a number of behaviours that are designed to modify physical characteristics of the character, such as its **joint stiffness** and **body damping**. These are: Hold; Body Damping; Body Stiffness; and Whole Body Stiffness. These behaviours do not general intelligent motion in themselves; rather they are used in conjunction with other behaviours to generate specific effects.

Hand and arm behaviours

There are a number of behaviours that are designed to affect the motion of the **hands** and **arms** of the simulation character. These are: Arms Crossed On Chest; Arms Raised Above Head; Arms Wide Of Head; Arms Windmill; Arms Zombie; Hands Covering Face; Hands From Behind Back; Hands Protecting Groin; and Hands Reach And Look At.

- **Arms Crossed On Chest** is an active behaviour. The character will try to move its arms in and across its chest. This behaviour is often used when you want the character to protect itself.
- **Arms Raised Above Head** is an active behaviour. The character will try to raise its arms and stretch them out in an upwards direction.
- **Arms Wide Of Head** is an active behaviour. The character will try to raise its arms and stretch them out in a sideways direction.
- **Arms Windmill** is an active behaviour. The character will try to raise its arms and stretch them out in a sideways direction, and then move them in a rotating motion. This behaviour is often used when you want the character to attempt to balance.
- **Arms Zombie** is an active behaviour. The character will try to raise its arms and stretch them out in a frontal direction.
- **Hands Covering Face** is an active behaviour. The character will try to move its arms in and across its face. This behaviour is often used when you want the character to protect itself.
- **Hands From Behind Back** is a conditional behaviour. The character will try to move its arms from behind its back to its sides. This behaviour is conditional because it will only be triggered if the character's arms are behind its back. This behaviour is often used when you want to move the character's arms into a reasonably neutral position before applying other behaviours or active poses.
- **Hands Protecting Groin** is an active behaviour. The character will try to move its arms down and across its groin region. This behaviour is often used when you want the character to protect itself.
- **Hands Reach And Look At** is an active behaviour that requires **target objects**. You can specify a different target object for each arm, and also for the head. If a given arm has a target object, the character will try and reach that arm out in the direction of the target. If the head has a target object, the character will try and look around in the direction of the target.

Arms Crossed On Chest

Arms Crossed On Chest is an active behaviour. The character will try to move its arms in and across its chest. This behaviour is often used when you want the character to protect itself. The behaviour variables for this behaviour are:

VARIABLES

Strength

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative power applied to drive the arm joints. With minimum strength, arms will reach chest only if unobstructed and in zero gravity. With maximum strength, arms will reach chest regardless of most obstacles.

Speed

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative rate at which the arms will approach the chest. With minimum speed, arms will reach chest in around 500 frames. With maximum speed, arms will reach chest in around 20 frames.

Leg and torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative capacity of the legs and torso to maintain their initial pose during the behaviour event. With minimum stiffness, the legs and torso will become limp. With maximum stiffness, the legs and torso will remain rigid.

Noise level

Range: 0.0 to 1.0. **Default value:** 0.0.

Specifies the relative amount of randomness applied to the behaviour. With minimum noise level, no noise is applied to the behaviour. With maximum noise level, maximum noise is applied. Adding randomness to a behaviour is a good way of generating more realistic animation.

Random number seed

Range: 1 to 10000. **Default value:** Random.

Establishes a sequence of random numbers used to add randomness, or noise, to the behaviour. Adding noise can add an additional level of realism to a behaviour. A particular random number seed will establish a repeatable noise sequence.

Arms Raised Above Head

Arms Raised Above Head is an active behaviour. The character will try to raise its arms and stretch them out in an upwards direction. The behaviour variables for this behaviour are:

VARIABLES

Speed

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative rate at which the arms will be raised. With minimum speed, arms will be raised in around 500 frames. With maximum speed, arms will be raised in around 20 frames.

Strength

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative power applied to drive the arm joints. With minimum strength, arms will be raised only if unobstructed and in zero gravity. With maximum strength, arms will be raised regardless of most obstacles.

Leg and torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative capacity of the legs and torso to maintain their initial pose during the behaviour event. With minimum stiffness, the legs and torso will become limp. With maximum stiffness, the legs and torso will remain rigid.

Noise level

Range: 0.0 to 1.0. **Default value:** 0.17.

Specifies the relative amount of randomness applied to the behaviour. With minimum noise level, no noise is applied to the behaviour. With maximum noise level, maximum noise is applied. Adding randomness to a behaviour is a good way of generating more realistic animation.

Random number seed

Range: 1 to 10000. **Default value:** Random.

Establishes a sequence of random numbers used to add randomness, or noise, to the behaviour. Adding noise can add an additional level of realism to a behaviour. A particular random number seed will establish a repeatable noise sequence.

Arms Wide Of Head

Arms Wide Of Head is an active behaviour. The character will try to raise its arms and stretch them out in a sideways direction. The behaviour variables for this behaviour are:

VARIABLES

Speed

Range: 0.0 to 1.0. **Default value:** 0.7.

Specifies the relative rate at which the arms will be moved sideways. With minimum speed, arms will be moved sideways in around 500 frames. With maximum speed, arms will be raised in around 20 frames.

Strength

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative power applied to drive the arm joints. With minimum strength, arms will be moved sideways only if unobstructed and in zero gravity. With maximum strength, arms will be moved sideways regardless of most obstacles.

Leg and torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative capacity of the legs and torso to maintain their initial pose during the behaviour event. With minimum stiffness, the legs and torso will become limp. With maximum stiffness, the legs and torso will remain rigid.

Noise level

Range: 0.0 to 1.0. **Default value:** 0.0.

Specifies the relative amount of randomness applied to the behaviour. With minimum noise level, no noise is applied to the behaviour. With maximum noise level, maximum noise is applied. Adding randomness to a behaviour is a good way of generating more realistic animation.

Random number seed

Range: 1 to 10000. **Default value:** Random.

Establishes a sequence of random numbers used to add randomness, or noise, to the behaviour. Adding noise can add an additional level of realism to a behaviour. A particular random number seed will establish a repeatable noise sequence.

Arms Windmill

Arms Windmill is an active behaviour. The character will try to raise its arms and stretch them out in a sideways direction, and then move them in a rotating motion. This behaviour is often used when you want the character to attempt to balance. The behaviour variables for this behaviour are:

VARIABLES

Shoulder amplitude

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative extent to which the arms will swing from the shoulders. With minimum amplitude, the arms will not swing. With maximum amplitude, the arms will swing wildly.

Elbow amplitude

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative extent to which the forearms will swing from the elbows. With minimum amplitude, the elbows will remain rigid. With maximum amplitude, the elbows will swing fully.

Velocity

Range: -1.0 to 1.0. **Default value:** +0.8.

Specifies the relative rate at which the arms swing. With minimum velocity, the arms will swing slowly and will complete a cycle in around 500 frames, if unobstructed. With maximum velocity, the arms will swing rapidly and will complete a cycle in around 20 frames, if unobstructed.

Synchronisation offset

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative offset at which the left and right arms will cycle with each other. With minimum offset, the left and right arms will rotate together, so that when one arm is fully forward, the other arm will also be fully forward. With maximum offset, the arms will cycle in opposition to each other, so that when one arm is fully forward, the other arm will be fully backwards.

Phase offset

Range: -1.0 to 1.0. **Default value:** 0.0.

Specifies the starting point for the oscillations that drive the arms. Zero offset implies an ideal starting point. A maximum negative offset implies the arms oscillate one cycle behind the ideal oscillation. A maximum positive offset implies the arms oscillate one cycle ahead of the ideal oscillation.

Strength

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative power applied to drive the arm joints. With minimum strength, arms will swing only if unobstructed and in zero gravity. With maximum strength, arms will swing regardless of most obstacles.

YX plane offset

Range: -1.0 to 1.0. **Default value:** +0.1.

Specifies the horizontal midpoint of the arm swing. Zero offset implies the arms swing to the side of the body. A maximum negative offset implies the arms will swing in front of the body. A maximum positive offset implies the arms will swing behind the body.

ZX plane offset

Range: -1.0 to 1.0. **Default value:** 0.0.

Specifies the vertical midpoint of the arm swing. Zero offset implies the arms swing at shoulder height. A maximum negative offset implies the arms will swing towards waist height. A maximum positive offset implies the arms will above the head height.

Leg and torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative capacity of the legs and torso to maintain their initial pose during the behaviour event. With minimum stiffness, the legs and torso will become limp. With maximum stiffness, the legs and torso will remain rigid.

Arms Zombie

Arms Zombie is an active behaviour. The character will try to raise its arms and stretch them out in a frontal direction. The behaviour variables for this behaviour are:

VARIABLES

Speed

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative rate at which the arms will be moved forward. With minimum speed, arms will be moved forward in around 500 frames. With maximum speed, arms will be moved forward in around 20 frames.

Strength

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative power applied to drive the arm joints. With minimum strength, arms will be moved forward only if unobstructed and in zero gravity. With maximum strength, arms will be moved forward regardless of most obstacles.

Leg and torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative capacity of the legs and torso to maintain their initial pose during the behaviour event. With minimum stiffness, the legs and torso will become limp. With maximum stiffness, the legs and torso will remain rigid.

Noise level

Range: 0.0 to 1.0. **Default value:** 0.0.

Specifies the relative amount of randomness applied to the behaviour. With minimum noise level, no noise is applied to the behaviour. With maximum noise level, maximum noise is applied. Adding randomness to a behaviour is a good way of generating more realistic animation.

Random number seed

Range: 1 to 10000. **Default value:** Random.

Establishes a sequence of random numbers used to add randomness, or noise, to the behaviour. Adding noise can add an additional level of realism to a behaviour. A particular random number seed will establish a repeatable noise sequence.

Hands Covering Face

Hands Covering Face is an active behaviour. The character will try to move its arms in and across its face. This behaviour is often used when you want the character to protect itself. The behaviour variables for this behaviour are:

VARIABLES

Speed

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative rate at which the arms will reach the head. With minimum speed, arms will reach the head in around 500 frames. With maximum speed, arms will reach the head in around 20 frames.

Strength

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative power applied to drive the arm joints. With minimum strength, arms will reach the head only if unobstructed and in zero gravity. With maximum strength, arms will reach the head regardless of most obstacles.

Leg and torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative capacity of the legs and torso to maintain their initial pose during the behaviour event. With minimum stiffness, the legs and torso will become limp. With maximum stiffness, the legs and torso will remain rigid.

Noise level

Range: 0.0 to 1.0. **Default value:** 0.0.

Specifies the relative amount of randomness applied to the behaviour. With minimum noise level, no noise is applied to the behaviour. With maximum noise level, maximum noise is applied. Adding randomness to a behaviour is a good way of generating more realistic animation.

Random number seed

Range: 1 to 10000. **Default value:** Random.

Establishes a sequence of random numbers used to add randomness, or noise, to the behaviour. Adding noise can add an additional level of realism to a behaviour. A particular random number seed will establish a repeatable noise sequence.

Hands From Behind Back

Hands From Behind Back is a conditional behaviour. The character will try to move its arms from behind its back to its sides. This behaviour is conditional because it will only be triggered if the character's arms are behind its back. This behaviour is often used when you want to move the character's arms into a reasonably neutral position before applying other behaviours or active poses. The behaviour variables for this behaviour are:

VARIABLES

Hip width scale

Range: 0.0 to 1.0. **Default value:** 0.4.

Specifies the hip width of the character relative to the standard simulation character. With minimum scale, the hip width is assumed to match the standard simulation character. With maximum scale, the hips are assumed to be substantially broader.

Speed

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative rate at which the arms will reach the head. With minimum speed, arms will be moved into position in around 500 frames. With maximum speed, arms will be moved into position in around 20 frames.

Strength

Range: 0.0 to 1.0. **Default value:** 0.4.

Specifies the relative power applied to drive the arm joints. With minimum strength, arms will move from behind the back only if unobstructed and in zero gravity. With maximum strength, arms will move from behind the back regardless of most obstacles.

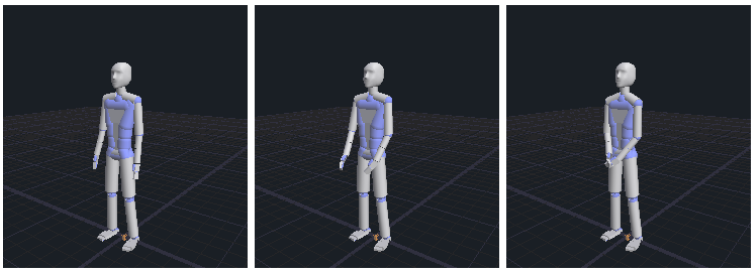
Leg and torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative capacity of the legs and torso to maintain their initial pose during the behaviour event. With minimum stiffness, the legs and torso will become limp. With maximum stiffness, the legs and torso will remain rigid.

Hands Protecting Groin

Hands Protecting Groin is an active behaviour. The character will try to move its arms down and across its groin region. This behaviour is often used when you want the character to protect itself. The behaviour variables for this behaviour are:



VARIABLES

Speed

Range: 0.0 to 1.0. **Default value:** 0.65.

Specifies the relative rate at which the arms will move toward the groin. With minimum speed, arms will be moved into position in around 500 frames. With maximum speed, arms will be moved into position in around 20 frames.

Strength

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative power applied to drive the arm joints. With minimum strength, arms will move towards the groin only if unobstructed and in zero gravity. With maximum strength, arms will move towards the groin regardless of most obstacles.

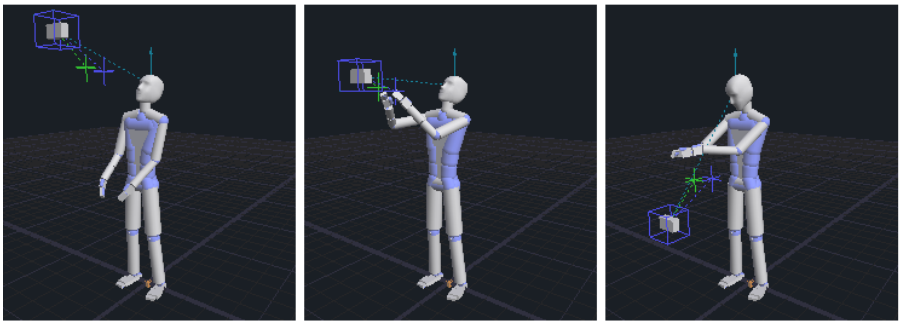
Leg and torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative capacity of the legs and torso to maintain their initial pose during the behaviour event. With minimum stiffness, the legs and torso will become limp. With maximum stiffness, the legs and torso will remain rigid.

Hands Reach And Look At

Hands Reach And Look At is an active behaviour that requires **target objects**. You can specify a different target object for each arm, and also for the head. If a given arm has a target object, the character will try and reach that arm out in the direction of the target. If the head has a target object, the character will try and look around in the direction of the target.



VARIABLES

Leg and torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative capacity of the legs and torso to maintain their initial pose during the behaviour event. With minimum stiffness, the legs and torso will become limp. With maximum stiffness, the legs and torso will remain rigid.

VARIABLES: Look At Target

Look at target

Range: Set of mass objects. **Default value:** Empty.

Specifies the mass object or objects that the head will track. If multiple mass objects are specified, the head will track the centre of mass of the objects. If no mass objects are selected, the head will not track.

Delay

Range: 0.0 to 1000.0. **Default value:** 0.0.

Specifies the time, in seconds, until the head begins to track the Look At target.

Blend

Range: 0.0 to 1000.0. **Default value:** 0.0.

Specifies the time, in seconds, until the neck joint power reaches its maximum. The blend setting affects the relative acceleration of the head as it tracks the Look At target.

Speed

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative rate at which the head tracks the Look At target. With minimum speed, the head will track the Look At target slowly. With maximum speed, the head will track the Look At target rapidly.

VARIABLES: Left Hand Target

Left hand target

Range: Set of mass objects. **Default value:** Empty.

Specifies the mass object or objects that the left hand will track. If multiple mass objects are specified, the left hand will track the centre of mass of the objects. If no mass objects are selected, the left hand will not track.

Speed

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative rate at which the left hand will approach the Left Hand target. With minimum speed, the left hand will track the Left Hand target slowly. With maximum speed, the left hand will track the Left Hand target rapidly.

Slowdown

Range: 0.0 to 1.0. **Default value:** 0.25.

Specifies the relative deceleration of the left hand as it approaches the Left Hand target. With minimum slowdown, the hand will not decelerate as it approaches the Left Hand target. With maximum slowdown, there will be a visible deceleration that commences when the hand is still some distance from the Left Hand target.

Delay

Range: 0.0 to 1000.0. **Default value:** 0.0.

Specifies the time, in seconds, until the left hand begins to track the Left Hand target.

Blend

Range: 0.0 to 1000.0. **Default value:** 0.0.

Specifies the time, in seconds, until the arm joint power reaches its maximum. The blend setting affects the relative acceleration of the left arm as it tracks the Left Hand target.

Strength

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative power applied to drive the left arm joints. With minimum strength, the left arm will reach the Left Hand target only if unobstructed and in zero gravity. With maximum strength, the left arm will reach the Left Hand target regardless of most obstacles.

Palm facing

Range: On/Off. **Default value:** On.

Specifies whether the character should orient its left hand so that the palm faces towards the Left Hand target. When palm facing is on, the character makes small corrective adjustments to its left wrist.

Orientation

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative direction with which the left hand will approach the Left Hand target. With minimum orientation, the left hand will approach the Left Hand target from below. With maximum orientation, the left hand will approach the Left Hand target from above.

Exaggeration

Range: 0.0 to 1.0. **Default value:** 0.0.

Specifies the relative path of the left hand as it approaches the Left Hand target. With minimum exaggeration, the path of the hand will be approximately linear. With maximum exaggeration, the path of the hand will be a large arc.

Maximum arm extension

Range: 0.0 to 1.0. **Default value:** 1.0.

Specifies the relative extension of the left arm as the left hand approaches the Left Hand target. With minimum extension, the left shoulder and elbow of the character will remain tucked close to its body. With maximum extension, the left shoulder and elbow will be relaxed so that the arm is free to extend as far as possible.

Fingers angle

Range: 0.0 to 1.0. **Default value:** 0.2.

Specifies the amount of finger rotation as the left hand approaches the Left Hand target. With minimum angle, the fingers will remain straight, and slightly overextended. With maximum angle, the fingers will be tucked in tightly so that the left hand forms a fist.

Use prediction

Range: On/Off. **Default value:** On.

Specifies whether the character should attempt to predict the motion of the Left Hand target. When prediction is off, the character moves its left arm to continually follow the path of the Left Hand target. When prediction is on, the character will move its left arm to attempt to intercept the Left Hand target, based on the trajectory of the target. Predictions are made at time intervals, based on the prediction rate. In addition, if the character is accelerating rapidly, or the Left Hand target is accelerating rapidly, additional predictions are made.

Prediction rate

Range: 0.0 to 10.0. **Default value:** 0.2.

Specifies the time period, in seconds, between predictions of the Left Hand target trajectory. A prediction rate of zero implies a new prediction at every frame of the simulation. Only used when prediction is turned on.

Target acceleration threshold

Range: 0.0 to 50.0. **Default value:** 0.05.

Specifies the relative acceleration of the Left Hand target required for a new prediction of its trajectory. A minimum threshold implies new predictions will be triggered at every frame. A maximum threshold implies that only very large changes in the Left Hand target acceleration will trigger a new prediction.

Character acceleration threshold

Range: 0.0 to 50.0. **Default value:** 20.0.

Specifies the relative acceleration of the character required for a new prediction of the Left Hand target trajectory. A minimum threshold implies new predictions will be triggered at every frame. A maximum threshold implies that only very large changes in the character acceleration will trigger a new prediction.

Trajectory offset

Range: -1.0 to +1.0. **Default value:** 0.0.

Specifies the relative position along the Left Hand target trajectory that the character will reach for. A zero offset implies the character will reach for the Left Hand target when it is as close as possible. A negative offset implies that the character will reach for the Left Hand target at an earlier time than this. A positive offset implies that the character will reach for the Left Hand target at a later time than this.

Interception offset X

Range: -10.0 to +10.0. **Default value:** 0.0.

Specifies an offset, along the global X direction, to the calculated Left Hand target position. Use this offset to make small adjustments to the position that the character reaches its left arm towards.

Interception offset Y

Range: -10.0 to +10.0. **Default value:** 0.0.

Specifies an offset, along the global Y direction, to the calculated Left Hand target position. Use this offset to make small adjustments to the position that the character reaches its left arm towards.

Interception offset Z

Range: -10.0 to +10.0. **Default value:** 0.0.

Specifies an offset, along the global Z direction, to the calculated Left Hand target position. Use this offset to make small adjustments to the position that the character reaches its left arm towards.

VARIABLES: Right Hand Target

Right hand target

Range: Set of mass objects. **Default value:** Empty.

Specifies the mass object or objects that the right hand will track. If multiple mass objects are specified, the right hand will track the centre of mass of the objects. If no mass objects are selected, the right hand will not track.

Speed

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative rate at which the right hand will approach the Right Hand target. With minimum speed, the right hand will track the Right Hand target slowly. With maximum speed, the right hand will track the Right Hand target rapidly.

Slowdown

Range: 0.0 to 1.0. **Default value:** 0.25.

Specifies the relative deceleration of the right hand as it approaches the Right Hand target. With minimum slowdown, the hand will not decelerate as it approaches the Right Hand target. With maximum slowdown, there will be a visible deceleration that commences when the hand is still some distance from the Right Hand target.

Delay

Range: 0.0 to 1000.0. **Default value:** 0.0.

Specifies the time, in seconds, until the right hand begins to track the Right Hand target.

Blend

Range: 0.0 to 1000.0. **Default value:** 0.0.

Specifies the time, in seconds, until the arm joint power reaches its maximum. The blend setting affects the relative acceleration of the right arm as it tracks the Right Hand target.

Strength

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative power applied to drive the right arm joints. With minimum strength, the right arm will reach the Right Hand target only if unobstructed and in zero gravity. With maximum strength, the right arm will reach the Right Hand target regardless of most obstacles.

Palm facing

Range: On/Off. **Default value:** On.

Specifies whether the character should orient its right hand so that the palm faces towards the Right Hand target. When palm facing is on, the character makes small corrective adjustments to its right wrist.

Orientation

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative direction with which the right hand will approach the Right Hand target. With minimum orientation, the right hand will approach the Right Hand target from below. With maximum orientation, the right hand will approach the Right Hand target from above.

Exaggeration

Range: 0.0 to 1.0. **Default value:** 0.0.

Specifies the relative path of the right hand as it approaches the Right Hand target. With minimum exaggeration, the path of the hand will be approximately linear. With maximum exaggeration, the path of the hand will be a large arc.

Maximum arm extension

Range: 0.0 to 1.0. **Default value:** 1.0.

Specifies the relative extension of the right arm as the right hand approaches the Right Hand target. With minimum extension, the right shoulder and elbow of the character will remain tucked close to its body. With maximum extension, the right shoulder and elbow will be relaxed so that the arm is free to extend as far as possible.

Fingers angle

Range: 0.0 to 1.0. **Default value:** 0.2.

Specifies the amount of finger rotation as the right hand approaches the Right Hand target. With minimum angle, the fingers will remain straight, and slightly overextended. With maximum angle, the fingers will be tucked in tightly so that the right hand forms a fist.

Use prediction

Range: On/Off. **Default value:** On.

Specifies whether the character should attempt to predict the motion of the Right Hand target. When prediction is off, the character moves its right arm to continually follow the path of the Right Hand target. When prediction is on, the character will move its right arm to attempt to intercept the Right Hand target, based on the trajectory of the target. Predictions are made at time intervals, based on the prediction rate. In addition, if the character is accelerating rapidly, or the Right Hand target is accelerating rapidly, additional predictions are made.

Prediction rate

Range: 0.0 to 10.0. **Default value:** 0.2.

Specifies the time period, in seconds, between predictions of the Right Hand target trajectory. A prediction rate of zero implies a new prediction at every frame of the simulation. Only used when prediction is turned on.

Target acceleration threshold

Range: 0.0 to 50.0. **Default value:** 0.05.

Specifies the relative acceleration of the Right Hand target required for a new prediction of its trajectory. A minimum threshold implies new predictions will be triggered at every frame. A maximum threshold implies that only very large changes in the Right Hand target acceleration will trigger a new prediction.

Character acceleration threshold

Range: 0.0 to 50.0. **Default value:** 20.0.

Specifies the relative acceleration of the character required for a new prediction of the Right Hand target trajectory. A minimum threshold implies new predictions will be triggered at every frame. A maximum threshold implies that only very large changes in the character acceleration will trigger a new prediction.

Trajectory offset

Range: -1.0 to +1.0. **Default value:** 0.0.

Specifies the relative position along the Right Hand target trajectory that the character will reach for. A zero offset implies the character will reach for the Right Hand target when it is as close as possible. A negative offset implies that the character will reach for the Right Hand target at an earlier time than this. A positive offset implies that the character will reach for the Right Hand target at a later time than this.

Interception offset X

Range: -10.0 to +10.0. **Default value:** 0.0.

Specifies an offset, along the global X direction, to the calculated Right Hand target position. Use this offset to make small adjustments to the position that the character reaches its right arm towards.

Interception offset Y

Range: -10.0 to +10.0. **Default value:** 0.0.

Specifies an offset, along the global Y direction, to the calculated Right Hand target position. Use this offset to make small adjustments to the position that the character reaches its right arm towards.

Interception offset Z

Range: -10.0 to +10.0. **Default value:** 0.0.

Specifies an offset, along the global Z direction, to the calculated Right Hand target position. Use this offset to make small adjustments to the position that the character reaches its right arm towards.

Leg behaviours

There are a number of behaviours that are designed to affect the motion of the **legs** of the simulation character. These are: Legs Kick; Legs Reach; and Legs Straighten.

- **Legs Kick** is an active behaviour. The character will try to move its legs in a kicking motion.
- **Legs Reach** is an active behaviour that requires **target objects**. You can specify a different target object for each leg. If a given leg has a target object, the character will try to reach that leg out in the direction of the target.
- **Legs Straighten** is an active behaviour. The character will try to straighten its legs.

Legs Kick

Legs Kick is an active behaviour. The character will try to move its legs in a kicking motion. The behaviour variables for this behaviour are:

VARIABLES

Hip amplitude

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative extent to which the legs will swing from the hips. With minimum amplitude, the legs will not swing. With maximum amplitude, the legs will swing to their maximum extent.

Knee amplitude

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative extent to which the lower legs will swing from the knees. With minimum amplitude, the knees will remain rigid. With maximum amplitude, the knees will swing to their maximum extent.

Velocity

Range: -1.0 to +1.0. **Default value:** 0.6.

Specifies the relative rate at which the legs will kick. With minimum velocity, the legs will kick slowly. With maximum velocity, the legs will kick rapidly.

Phase offset

Range: -1.0 to +1.0. **Default value:** 0.0.

Specifies the starting point for the oscillations that drive the leg kicks. Zero offset implies an ideal starting point. A maximum negative offset implies the leg kicking oscillates one cycle behind the ideal oscillation. A maximum positive offset implies the leg kicking oscillates one cycle ahead of the ideal oscillation.

Strength

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative power applied to drive the leg joints. With minimum strength, the legs will kick only if unobstructed and in zero gravity. With maximum strength, the legs will kick regardless of most obstacles.

Arms stiffness

Range: 0.0 to 1.0. **Default value:** 0.2.

Specifies the relative capacity of the arms to maintain their initial pose during the behaviour event. With minimum stiffness, the arms will become limp. With maximum stiffness, the arms will remain rigid.

Torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative capacity of the torso to maintain its initial pose during the behaviour event. With minimum stiffness, the torso will become limp. With maximum stiffness, the torso will remain rigid.

Noise level

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative amount of randomness applied to the behaviour. With minimum noise level, no noise is applied to the behaviour. With maximum noise level, maximum noise is applied. Adding randomness to a behaviour is a good way of generating more realistic animation.

Random number seed

Range: 1 to 10000. **Default value:** Random.

Establishes a sequence of random numbers used to add randomness, or noise, to the behaviour. Adding noise can add an additional level of realism to a behaviour. A particular random number seed will establish a repeatable noise sequence.

Legs Reach

Legs Reach is an active behaviour that requires **target objects**. You can specify a different target object for each leg. If a given leg has a target object, the character will try to reach that leg out in the direction of the target. The behaviour variables for this behaviour are:

VARIABLES

Arms and torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative capacity of the arms and torso to maintain their initial pose during the behaviour event. With minimum stiffness, the arms and torso will become limp. With maximum stiffness, the arms and torso will remain rigid.

VARIABLES: Left Leg Target

Left leg target

Range: Set of mass objects. **Default value:** Empty.

Specifies the mass object or objects that the left leg will track. If multiple mass objects are specified, the left leg will track the centre of mass of the objects. If no mass objects are selected, the left leg will not track.

Speed

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative rate at which the left leg will approach the Left Leg target. With minimum speed, the left leg will track the Left Leg target slowly. With maximum speed, the left leg will track the Left Leg target rapidly.

Slowdown

Range: 0.0 to 1.0. **Default value:** 0.25.

Specifies the relative deceleration of the left leg as it approaches the Left Leg target. With minimum slowdown, the leg will not decelerate as it approaches the Left Leg target. With maximum slowdown, there will be a visible deceleration that commences when the leg is still some distance from the Left Leg target.

Strength

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative power applied to drive the left leg joints. With minimum strength, the left leg will reach the Left Leg target only if unobstructed and in zero gravity. With maximum strength, the left leg will reach the Left Leg target regardless of most obstacles.

Blend

Range: 0.0 to 1000.0. **Default value:** 0.0.

Specifies the time, in seconds, until the left leg joint power reaches its maximum. The blend setting affects the relative acceleration of the left leg as it tracks the Left Leg target.

Delay

Range: 0.0 to 1000.0. **Default value:** 0.0.

Specifies the time, in seconds, until the left leg begins to track the Left Leg target.

Maximum leg extension

Range: 0.0 to 1.0. **Default value:** 1.0.

Specifies the relative extension of the left leg as the left foot approaches the Left Leg target. With minimum extension, the left hip and knee of the character will remain tucked close to its body. With maximum extension, the left knee and hip will be relaxed so that the leg is free to extend as far as possible.

Use prediction

Range: On/Off. **Default value:** On.

Specifies whether the character should attempt to predict the motion of the Left Leg target. When prediction is off, the character moves its left leg to continually follow the path of the Left Leg target. When prediction is on, the character will move its left leg to attempt to intercept the Left Leg target, based on the trajectory of the target. Predictions are made at time intervals, based on the prediction rate. In addition, if the character is accelerating rapidly, or the Left Leg target is accelerating rapidly, additional predictions are made.

Prediction rate

Range: 0.0 to 10.0. **Default value:** 0.5.

Specifies the time period, in seconds, between predictions of the Left Leg target trajectory. A prediction rate of zero implies a new prediction at every frame of the simulation. Only used when prediction is turned on.

Character acceleration threshold

Range: 0.0 to 50.0. **Default value:** 20.0.

Specifies the relative acceleration of the character required for a new prediction of the Left Leg target trajectory. A minimum threshold implies new predictions will be triggered at every frame. A maximum threshold implies that only very large changes in the character acceleration will trigger a new prediction.

Target acceleration threshold

Range: 0.0 to 50.0. **Default value:** 0.1.

Specifies the relative acceleration of the Left Leg target required for a new prediction of its trajectory. A minimum threshold implies new predictions will be triggered at every frame. A maximum threshold implies that only very large changes in the Left Leg target acceleration will trigger a new prediction.

Trajectory offset

Range: -1.0 to +1.0. **Default value:** 0.0.

Specifies the relative position along the Left Leg target trajectory that the character will reach for. A zero offset implies the character will reach for the Left Leg target when it is as close as possible. A negative offset implies that the character will reach for the Left Leg target at an earlier time than this. A positive offset implies that the character will reach for the Left Leg target at a later time than this.

Interception offset X

Range: -10.0 to +10.0. **Default value:** 0.0.

Specifies on offset, along the global X direction, to the calculated Left Leg target position. Use this offset to make small adjustments to the position that the character reaches its left foot towards.

Interception offset Y

Range: -10.0 to +10.0. **Default value:** 0.0.

Specifies on offset, along the global Y direction, to the calculated Left Leg target position. Use this offset to make small adjustments to the position that the character reaches its left foot towards.

Interception offset Z

Range: -10.0 to +10.0. **Default value:** 0.0.

Specifies on offset, along the global Z direction, to the calculated Left Leg target position. Use this offset to make small adjustments to the position that the character reaches its left foot towards.

VARIABLES: Right Leg Target

Right leg target

Range: Set of mass objects. **Default value:** Empty.

Specifies the mass object or objects that the right leg will track. If multiple mass objects are specified, the right leg will track the centre of mass of the objects. If no mass objects are selected, the right leg will not track.

Speed

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative rate at which the right leg will approach the Right Leg target. With minimum speed, the right leg will track the Right Leg target slowly. With maximum speed, the right leg will track the Right Leg target rapidly.

Slowdown

Range: 0.0 to 1.0. **Default value:** 0.25.

Specifies the relative deceleration of the right leg as it approaches the Right Leg target. With minimum slowdown, the leg will not decelerate as it approaches the Right Leg target. With maximum slowdown, there will be a visible deceleration that commences when the hand is still some distance from the Right Leg target.

Strength

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative power applied to drive the right leg joints. With minimum strength, the right leg will reach the Right Leg target only if unobstructed and in zero gravity. With maximum strength, the right leg will reach the Right Leg target regardless of most obstacles.

Blend

Range: 0.0 to 1000.0. **Default value:** 0.0.

Specifies the time, in seconds, until the right leg joint power reaches its maximum. The blend setting affects the relative acceleration of the right leg as it tracks the Right Leg target.

Delay

Range: 0.0 to 1000.0. **Default value:** 0.0.

Specifies the time, in seconds, until the right leg begins to track the Right Leg target.

Maximum leg extension

Range: 0.0 to 1.0. **Default value:** 1.0.

Specifies the relative extension of the right leg as the right foot approaches the Right Leg target. With minimum extension, the right hip and knee of the character will remain tucked close to its body. With maximum extension, the right knee and hip will be relaxed so that the leg is free to extend as far as possible.

Use prediction

Range: On/Off. **Default value:** On.

Specifies whether the character should attempt to predict the motion of the Right Leg target. When prediction is off, the character moves its right leg to continually follow the path of the Right Leg target. When prediction is on, the character will move its right leg to attempt to intercept the Right Leg target, based on the trajectory of the target. Predictions are made at time intervals, based on the prediction rate. In addition, if the character is accelerating rapidly, or the Right Leg target is accelerating rapidly, additional predictions are made.

Prediction rate

Range: 0.0 to 10.0. **Default value:** 0.5.

Specifies the time period, in seconds, between predictions of the Right Leg target trajectory. A prediction rate of zero implies a new prediction at every frame of the simulation. Only used when prediction is turned on.

Character acceleration threshold

Range: 0.0 to 50.0. **Default value:** 20.0.

Specifies the relative acceleration of the character required for a new prediction of the Right Leg target trajectory. A minimum threshold implies new predictions will be triggered at every frame. A maximum threshold implies that only very large changes in the character acceleration will trigger a new prediction.

Target acceleration threshold

Range: 0.0 to 50.0. **Default value:** 0.1.

Specifies the relative acceleration of the Right Leg target required for a new prediction of its trajectory. A minimum threshold implies new predictions will be triggered at every frame. A maximum threshold implies that only very large changes in the Right Leg target acceleration will trigger a new prediction.

Trajectory offset

Range: -1.0 to +1.0. **Default value:** 0.0.

Specifies the relative position along the Right Leg target trajectory that the character will reach for. A zero offset implies the character will reach for the Right Leg target when it is as close as possible. A negative offset implies that the character will reach for the Right Leg target at an earlier time than this. A positive offset implies that the character will reach for the Right Leg target at a later time than this.

Interception offset X

Range: -10.0 to +10.0. **Default value:** 0.0.

Specifies an offset, along the global X direction, to the calculated Right Leg target position. Use this offset to make small adjustments to the position that the character reaches its right foot towards.

Interception offset Y

Range: -10.0 to +10.0. **Default value:** 0.0.

Specifies an offset, along the global Y direction, to the calculated Right Leg target position. Use this offset to make small adjustments to the position that the character reaches its right foot towards.

Interception offset Z

Range: -10.0 to +10.0. **Default value:** 0.0.

Specifies an offset, along the global Z direction, to the calculated Right Leg target position. Use this offset to make small adjustments to the position that the character reaches its right foot towards.

Legs Straighten

Legs Straighten is an active behaviour. The character will try to straighten its legs. The behaviour variables for this behaviour are:

VARIABLES

Straighten left leg

Range: On/Off. **Default value:** On.

Specifies whether to straighten the left leg.

Left leg speed

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative rate at which the left leg will straighten. With minimum speed, the left leg will straighten slowly. With maximum speed, the left leg will straighten rapidly.

Left leg strength

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative power applied to drive the leg joints. With minimum strength, the legs will straighten only if unobstructed and in zero gravity. With maximum strength, the legs will straighten regardless of most obstacles.

Straighten right leg

Range: On/Off. **Default value:** On.

Specifies whether to straighten the right leg.

Right leg speed

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative rate at which the right leg will straighten. With minimum speed, the right leg will straighten slowly. With maximum speed, the right leg will straighten rapidly.

Right leg strength

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative power applied to drive the leg joints. With minimum strength, the legs will straighten only if unobstructed and in zero gravity. With maximum strength, the legs will straighten regardless of most obstacles.

Arm and torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.3.

Specifies the relative capacity of the arms and torso to maintain their initial pose during the behaviour event. With minimum stiffness, the arms and torso will become limp. With maximum stiffness, the arms and torso will remain rigid.

Whole body behaviours

There are a number of behaviours that are designed to affect the motion of the **entire body** of the simulation character. They include Balance; Body Foetal; Catch Fall; Fall Back, Twist And Catch Fall; Fall Back, Twist And Cover Face; Fall Back, Twist With Passive Arms; Jump; Jump And Dive; Land And Crouch; Stagger; Tackle; Writhe; and Writhe In Mid-Air.

- **Balance** is a conditional behaviour. If the character has contact with the ground, it will try to keep itself balanced. This behaviour is conditional because it requires that the character's feet have a solid contact with the ground or another object, and also that the character is reasonably upright, before commencing the balance.
- **Balance With Props** is a conditional behaviour. It extends Balance to allow for the selection of props, enabling the character to balance while also carrying or holding objects.
- **Body Foetal** is an active behaviour. The character will try to curl into a ball by pulling its arms and legs in towards its torso. This behaviour is often used when you want the character to protect itself.
- **Catch Fall** is a conditional behaviour. If the character is falling towards the ground, it will try to catch its fall by moving its arms out to prevent its head and upper torso from impacting with the ground. This behaviour is conditional because it will only be triggered if the character is falling towards the ground.
- **Fall Back, Twist And Catch Fall** is a conditional behaviour. If the character is falling backwards towards the ground, it will try and twist around, and then try to catch its fall by moving its arms out to prevent its head and upper torso from impacting with the ground. This behaviour is conditional because it will only be triggered if the character is falling backwards towards the ground.
- **Fall Back, Twist And Cover Face** is a conditional behaviour. If the character is falling backwards towards the ground, it will try and twist around, and then try to catch its fall by moving its arms up to shield its head from impact with the ground. This behaviour is conditional because it will only be triggered if the character is falling backwards towards the ground.
- **Fall Back, Twist With Passive Arms** is a conditional behaviour. If the character is falling backwards towards the ground, it will try and twist around. This behaviour is conditional because it will only be triggered if the character is falling backwards towards the ground.
- **Jump** is a conditional behaviour. If the character has contact with the ground, it will try to jump forward to propel itself through the air. After jumping, the character will try and land on its feet. This behaviour is conditional because it requires that the character's feet have a solid contact with the ground or another object before commencing the jump.

- **Jump And Dive** is a conditional behaviour. If the character has contact with the ground and is moving forward, it will try to jump forward to propel itself through the air. After jumping, the character will try and land horizontally. This behaviour is conditional because it requires that the character's feet have a solid contact with the ground or another object, and also that the character has some forward momentum, before commencing the jump.
- **Land And Crouch** is a conditional behaviour. If the character is falling towards the ground, it will wave its arms sideways, before bracing itself ahead of impact with the ground. After impact, the character will try and catch its fall by using its arms. This behaviour is conditional because it requires that the character is initially airborne and roughly upright.
- **Stagger** is a conditional behaviour. The character is unbalanced, it will try and prevent itself falling to the ground by taking a series of protective steps, and also by using its arms to try and stay balanced. This behaviour is conditional because it requires that the character is initially unbalanced.
- **Tackle** is a conditional behaviour that requires a **target character**. If the character is roughly within an arm-length of the target character, it will try and wrap its arms around the target character. This behaviour is conditional because it requires that the two characters are within an arm-length of each other.
- **Writhe** is an active behaviour. The character will try to move its arms and legs in kicking or thrashing motion.
- **Writhe In Mid-Air** is a conditional behaviour. If the character is airborne, it will try to move its arms and legs in kicking or thrashing motion. This behaviour is conditional because it requires that the character is initially airborne.

Balance

Balance is a conditional behaviour. If the character has contact with the ground, it will try to keep itself balanced. This behaviour is conditional because it requires that the character's feet have a solid contact with the ground or another object, and also that the character is reasonably upright, before commencing the balance. The behaviour variables for this behaviour are:

VARIABLES

Centre of mass offset X

Range: -1.0 to +1.0. **Default value:** 0.0.

Specifies an offset, along the global X direction, to the optimal balancing position. Ordinarily, the character attempts to balance by moving its centre of mass towards an optimal balancing position. When this offset is non-zero, the optimal balancing position is shifted. Use this offset to generate specific balancing effects.

Centre of mass offset Z

Range: -1.0 to +1.0. **Default value:** 0.0.

Specifies an offset, along the global Y direction, to the optimal balancing position. Ordinarily, the character attempts to balance by moving its centre of mass towards an optimal balancing position. When this offset is non-zero, the optimal balancing position is shifted. Use this offset to generate specific balancing effects.

Pitch offset

Range: -1.0 to +1.0. **Default value:** 0.0.

Specifies an offset, along a line that passes through the character's centre of mass and perpendicular to the line joining the centres of its feet, to the optimal balancing position. Ordinarily, the character attempts to balance by moving its centre of mass towards an optimal balancing position. When this offset is non-zero, the optimal balancing position is shifted. Use this offset to generate specific balancing effects.

Pelvis height

Range: 0.0 to 1.0. **Default value:** 1.0.

Specifies the relative extent to which the character will attempt to keep its pelvis high and its legs straight while balancing. With minimum pelvis height, the character will attempt to balance with its knees bent as far as possible. With maximum pelvis height, the character will attempt to balance with its knees as straight as possible.

Use arms

Range: On/Off. **Default value:** On.

Specifies whether the character will use arm movement to help maintain balance.

Balance threshold

Range: 0.0 to 1.0. **Default value:** 0.15.

Specifies the relative amount of imbalance that the character will tolerate while attempting to balance. If the relative amount of imbalance exceeds the threshold, the character will transition into a stagger behaviour. With minimum balance threshold, the balance will readily transition to a stagger behaviour. With maximum balance threshold, the character will continue to attempt to balance until its feet lose contact with the ground.

Balance With Props

Balance With Props is a conditional behaviour. Extends Balance to allow for the selection and inclusion of props, enabling to character to balance while also carrying or holding objects. The behaviour variables for this behaviour are:

VARIABLES

Centre of mass offset X

Range: -1.0 to +1.0. **Default value:** 0.0.

Specifies an offset, along the global X direction, to the optimal balancing position. Ordinarily, the character attempts to balance by moving its centre of mass towards an optimal balancing position. When this offset is non-zero, the optimal balancing position is shifted. Use this offset to generate specific balancing effects.

Centre of mass offset Z

Range: -1.0 to +1.0. **Default value:** 0.0.

Specifies an offset, along the global Y direction, to the optimal balancing position. Ordinarily, the character attempts to balance by moving its centre of mass towards an optimal balancing position. When this offset is non-zero, the optimal balancing position is shifted. Use this offset to generate specific balancing effects.

Pitch offset

Range: -1.0 to +1.0. **Default value:** 0.0.

Specifies an offset, along a line that passes through the character's centre of mass and perpendicular to the line joining the centres of its feet, to the optimal balancing position. Ordinarily, the character attempts to balance by moving its centre of mass towards an optimal balancing position. When this offset is non-zero, the optimal balancing position is shifted. Use this offset to generate specific balancing effects.

Pelvis height

Range: 0.0 to 1.0. **Default value:** 1.0.

Specifies the relative extent to which the character will attempt to keep its pelvis high and its legs straight while balancing. With minimum pelvis height, the character will attempt to balance with its knees bent as far as possible. With maximum pelvis height, the character will attempt to balance with its knees as straight as possible.

Use arms

Range: On/Off. **Default value:** On.

Specifies whether the character will use arm movement to help maintain balance.

Balance threshold

Range: 0.0 to 1.0. **Default value:** 0.15.

Specifies the relative amount of imbalance that the character will tolerate while attempting to balance. If the relative amount of imbalance exceeds the threshold, the character will transition into a stagger behaviour. With minimum balance threshold, the balance will readily transition to a stagger behaviour. With maximum balance threshold, the character will continue to attempt to balance until its feet lose contact with the ground.

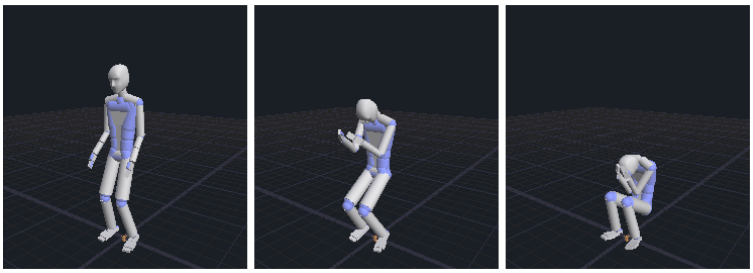
Props

Range: Set of mass objects. **Default value:** Empty.

User selected set of mass objects that the behaviour will take into account when attempting to balance. For this to work correctly, these objects will need to be either constrained to the character or attached in some other way.

Body Foetal

Body Foetal is an active behaviour. The character will try to curl into a ball by pulling its arms and legs in towards its torso. This behaviour is often used when you want the character to protect itself. The behaviour variables for this behaviour are:



VARIABLES

Torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative capacity of the torso to maintain its initial pose during the behaviour event. With minimum stiffness, the torso will become limp. With maximum stiffness, the torso will remain rigid.

Arms speed

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative rate at which the arms will tuck into the chest. With minimum speed, arms will tuck into the chest in around 500 frames. With maximum speed, arms will reach chest in around 20 frames.

Arms strength

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative power applied to drive the arm joints. With minimum strength, arms will tuck only if unobstructed and in zero gravity. With maximum strength, arms will tuck regardless of most obstacles.

Legs speed

Range: 0.0 to 1.0. **Default value:** 0.7.

Specifies the relative rate at which the legs tuck into the chest. With minimum speed, the legs will tuck slowly. With maximum speed, the legs will tuck quickly.

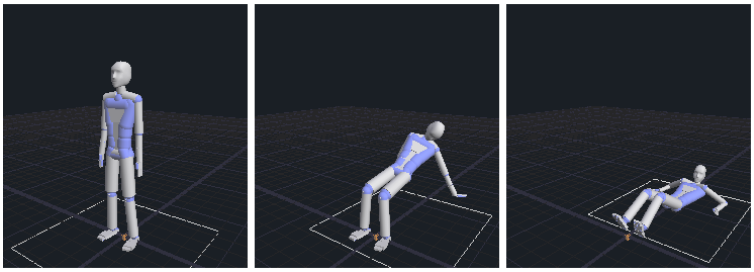
Legs strength

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative power applied to drive the leg joints. With minimum strength, legs will tuck only if unobstructed and in zero gravity. With maximum strength, legs will tuck regardless of most obstacles.

Catch Fall

Catch Fall is a conditional behaviour. If the character is falling towards the ground, it will try to catch its fall by moving its arms out to prevent its head and upper torso from impacting with the ground. This behaviour is conditional because it will only be triggered if the character is falling towards the ground. The behaviour variables for this behaviour are:



VARIABLES

Arms apart

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative separation of the arms as the character impacts with the ground. With minimum separation setting, the arms will be close together at the moment of impact. With maximum scale, the arms will be separated by approximately twice the shoulder width.

Arms speed

Range: 0.0 to 1.0. **Default value:** 0.7.

Specifies the relative rate at which the arms will reach out as the character approaches impact with the ground. With minimum speed, the arms will reach out slowly and it is probable that they will not have extended fully before the character impacts with the ground. With maximum speed, the arms will reach out rapidly and the impact will be cushioned successfully in most scenarios.

Arms strength

Range: 0.0 to 1.0. **Default value:** 0.7.

Specifies the relative power applied to drive the arm joints. With minimum strength, the arms will collapse on impact with the ground. With maximum strength, the arms will remain stiff on impact in most scenarios.

Predict ground height

Range: On/Off. **Default value:** On.

Specifies whether the character should assume the ground is located at the position of its lowest foot. If ground height prediction is turned off, the character uses the user-defined ground height setting.

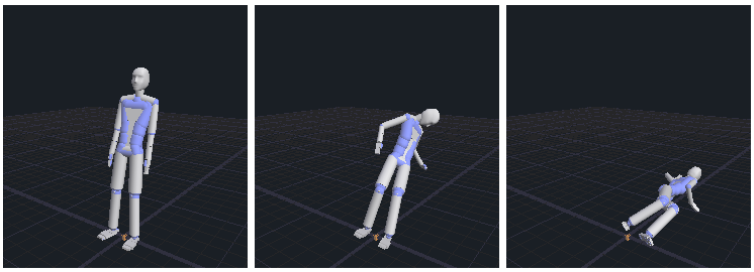
Ground height

Range: -100.0 to +100.0. **Default value:** 0.0.

Specifies a ground height relative to the position of the character. Only used if ground height prediction is turned off.

Fall Back, Twist And Catch Fall

Fall Back, Twist And Catch Fall is a conditional behaviour. If the character is falling backwards towards the ground, it will try and twist around, and then try to catch its fall by moving its arms out to prevent its head and upper torso from impacting with the ground. This behaviour is conditional because it will only be triggered if the character is falling backwards towards the ground. The behaviour variables for this behaviour are:



VARIABLES

Twist urgency

Range: 0.0 to 1.0. **Default value:** 0.3.

Specifies the relative speed with which the character attempts to reorient itself as it falls. With minimum urgency, the character will not reorient itself. With maximum urgency, the character will reorient itself rapidly.

Arms apart

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative separation of the arms as the character impacts with the ground. With minimum separation setting, the arms will be close together at the moment of impact. With maximum scale, the arms will be separated by approximately twice the shoulder width.

Arms speed

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative rate at which the arms will reach out as the character approaches impact with the ground. With minimum speed, the arms will reach out slowly and it is probable that they will not have extended fully before the character impacts with the ground. With maximum speed, the arms will reach out rapidly and the impact will be cushioned successfully in most scenarios.

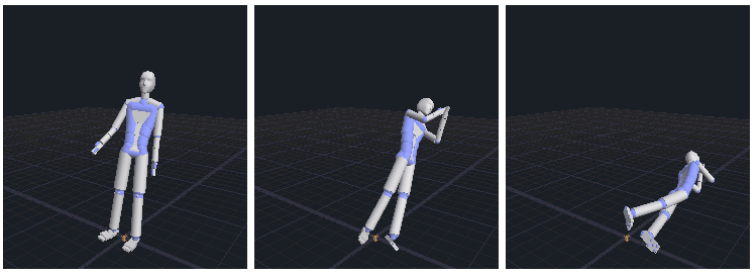
Arms strength

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative power applied to drive the arm joints. With minimum strength, the arms will collapse on impact with the ground. With maximum strength, the arms will remain stiff on impact in most scenarios.

Fall Back, Twist And Cover Face

Fall Back, Twist And Cover Face is a conditional behaviour. If the character is falling backwards towards the ground, it will try and twist around, and then try to catch its fall by moving its arms up to shield its head from impact with the ground. This behaviour is conditional because it will only be triggered if the character is falling backwards towards the ground. The behaviour variables for this behaviour are:



VARIABLES

Twist urgency

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative speed with which the character attempts to reorient itself as it falls. With minimum urgency, the character will not reorient itself. With maximum urgency, the character will reorient itself rapidly.

Arms speed

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative rate at which the arms will up towards its head as the character approaches impact with the ground. With minimum speed, the arms will not move towards the head. With maximum speed, the arms will reach up towards the head in around 20 frames.

Arms strength

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative power applied to drive the arm joints. With minimum strength, arms will reach the head only if unobstructed and in zero gravity. With maximum strength, arms will reach the head regardless of most obstacles.

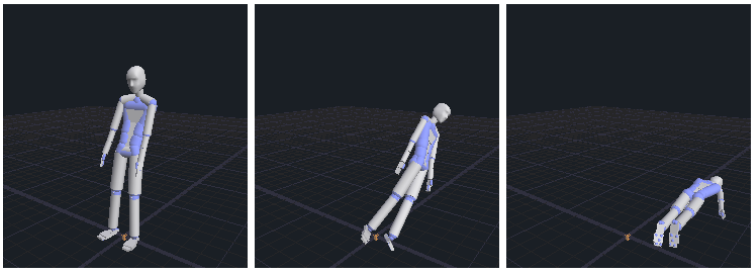
Leg and torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative capacity of the legs and torso to maintain their initial pose during the behaviour event. With minimum stiffness, the legs and torso will become limp. With maximum stiffness, the legs and torso will remain rigid.

Fall Back, Twist With Passive Arms

Fall Back, Twist With Passive Arms is a conditional behaviour. If the character is falling backwards towards the ground, it will try and twist around. This behaviour is conditional because it will only be triggered if the character is falling backwards towards the ground. The behaviour variables for this behaviour are:



VARIABLES

Twist urgency

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative speed with which the character attempts to reorient itself as it falls. With minimum urgency, the character will not reorient itself. With maximum urgency, the character will reorient itself rapidly.

Arms stiffness

Range: 0.0 to 1.0. **Default value:** 0.1.

Specifies the relative capacity of the arms to maintain their initial pose during the behaviour event. With minimum stiffness, the arms will become limp. With maximum stiffness, the arms will remain rigid.

Leg and torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative capacity of the legs and torso to maintain their initial pose during the behaviour event. With minimum stiffness, the legs and torso will become limp. With maximum stiffness, the legs and torso will remain rigid.

Jump

Jump is a conditional behaviour. If the character has contact with the ground, it will try to jump forward to propel itself through the air. After jumping, the character will try and land on its feet. This behaviour is conditional because it requires that the character's feet have a solid contact with the ground or another object before commencing the jump. The behaviour variables for this behaviour are:

VARIABLES

Strength

Range: 0.0 to 1.0. **Default value:** 0.4.

Specifies the relative power applied to drive the leg joints. With minimum strength, the character will have difficulty in losing contact with the ground. With maximum strength, the character will readily be able to lose contact with the ground.

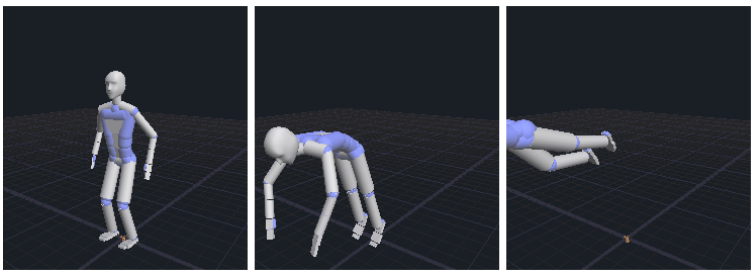
Jump timing

Range: 0.0 to 1.0. **Default value:** 0.3.

Specifies the relative time between the character bending its knees to prepare to jump, and the jump itself. With minimum timing, the character will briefly bend its knees before jumping. With maximum timing, the character will bend its knees more fully and will only jump when it starts to fall forward.

Jump And Dive

Jump And Dive is a conditional behaviour. If the character has contact with the ground and is moving forward, it will try to jump forward to propel itself through the air. After jumping, the character will try and land horizontally. This behaviour is conditional because it requires that the character's feet have a solid contact with the ground or another object, and also that the character has some forward momentum, before commencing the jump. The behaviour variables for this behaviour are:



VARIABLES

Strength

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative power applied to drive the leg joints. With minimum strength, the character will have difficulty in losing contact with the ground. With maximum strength, the character will readily be able to lose contact with the ground.

Jump timing

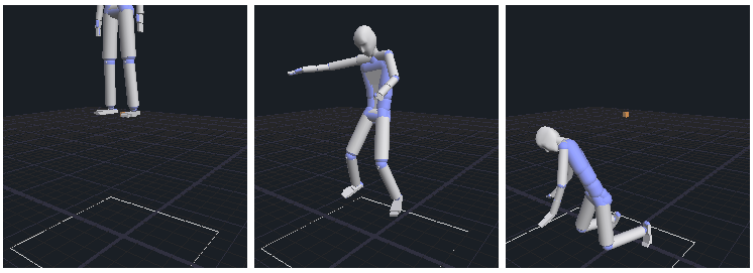
Range: 0.0 to 1.0. **Default value:** 0.75.

Specifies the relative time between the character bending its knees to prepare to jump, and the jump itself. With minimum timing, the character will briefly bend its knees before

jumping. With maximum timing, the character will bend its knees more fully and will only jump when it starts to fall forward.

Land And Crouch

Land And Crouch is a conditional behaviour. If the character is falling towards the ground, it will wave its arms sideways, before bracing itself ahead of impact with the ground. After impact, the character will try and catch its fall by using its arms. This behaviour is conditional because it requires that the character is initially airborne and roughly upright. The behaviour variables for this behaviour are:



VARIABLES

Legs strength

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative power applied to drive the leg joints. With minimum strength, the legs will collapse on impact with the ground. With maximum strength, the legs will remain stiff on impact in most scenarios.

Arms urgency

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative rate of arm movement while the character is airborne. With minimum urgency, the arms will undergo slow movements. With maximum urgency, the arms will undergo rapid movements.

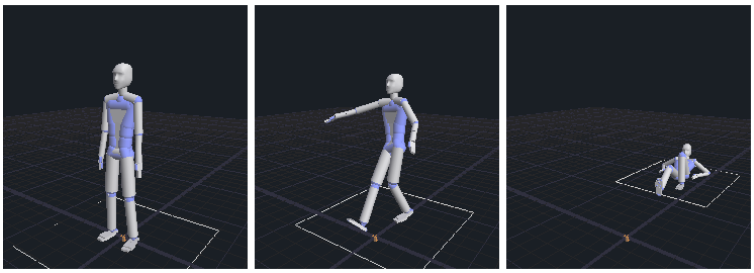
Ground height

Range: -100.0 to +100.0. **Default value:** 0.0.

Specifies a ground height relative to the position of the character.

Stagger

Stagger is a conditional behaviour. The character is unbalanced, it will try and prevent itself falling to the ground by taking a series of protective steps, and also by using its arms to try and stay balanced. This behaviour is conditional because it requires that the character is initially unbalanced. The behaviour variables for this behaviour are:



VARIABLES

Apply arms windmill

Range: On/Off. **Default value:** On.

Specifies whether the character will use arm movement to help maintain balance while staggering.

Stagger squarely

Range: On/Off. **Default value:** Off.

Specifies whether the character will try and maintain the forward direction of its pelvis while staggering. Turning this setting on can sometimes improve the ability of the character to maintain its balance while staggering.

Predict ground height

Range: On/Off. **Default value:** On.

Specifies whether the character should assume the ground is located at the position of its lowest foot. If ground height prediction is turned off, the character uses the user-defined ground height setting.

Ground height

Range: -100.0 to +100.0. **Default value:** 0.0.

Specifies a ground height relative to the position of the character. Only used if ground height prediction is turned off.

Maximum steps

Range: 1 to 20. **Default value:** 5.

Specifies the maximum number of steps the character will take when staggering before falling.

Arms speed

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative rate of arm movement while the character is staggering. With minimum speed, the arms will undergo slow movements. With maximum speed, the arms will undergo rapid movements.

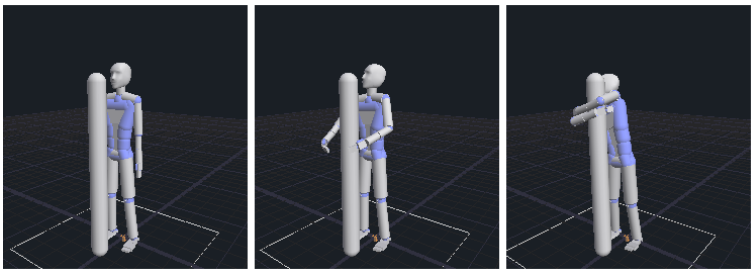
Arms strength

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative power applied to drive the arm joints. With minimum strength, arms will windmill only if unobstructed and in zero gravity. With maximum strength, arms will windmill regardless of most obstacles.

Tackle

Tackle is a conditional behaviour that requires a **tackle target character**. If the character is roughly within an arm-length of the target character, it will try and wrap its arms around the target character. This behaviour is conditional because it requires that the two characters are within an arm-length of each other, and also that the tackling character is moving quite rapidly in the direction of the tackle target character. The behaviour variables for this behaviour are:



VARIABLES

Arms strength

Range: 0.0 to 1.0. **Default value:** 0.8.

Specifies the relative power applied to drive the arm joints. With minimum strength, arms will reach to tackle only if unobstructed and in zero gravity. With maximum strength, arms will reach to tackle regardless of most obstacles.

Arms speed

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative rate of arm movement towards the tackle target character. With minimum speed, the character to will extend its arms slowly towards the tackle target character. With maximum speed, the character will extend its arms rapidly towards the tackle target character.

Leg and torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.5.

Specifies the relative capacity of the legs and torso to maintain their initial pose during the behaviour event. With minimum stiffness, the legs and torso will become limp. With maximum stiffness, the legs and torso will remain rigid.

Predict ground height

Range: On/Off. **Default value:** On.

Specifies whether the character should assume the ground is located at the position of its lowest foot. If ground height prediction is turned off, the character uses the user-defined ground height setting.

Ground height

Range: -100.0 to +100.0. **Default value:** 0.0.

Specifies a ground height relative to the position of the character. Only used if ground height prediction is turned off.

Reach forward duration

Range: 0.0 to 1000.0. **Default value:** 0.05.

Specifies the time, in seconds, during which the character reaches forward as it approaches the tackle target character.

Widen arms duration

Range: 0.0 to 1000.0. **Default value:** 0.0.

Specifies the time, in seconds, during which the character separates its arms as it approaches the tackle target character.

Upper spine target

Range: Set of collision or mass objects. **Default value:** Empty.

Specifies a set of collision objects on the tackle target character which represent an upper spine target for the tackling character. The upper spine target must be specified in order to use a tackle behaviour. Usually a single collision object is sufficient to define the upper spine target. You can specify one or more mass objects as the upper spine target, although be aware that mass object positions are not updated when the tackle target character is in the Collision Only simulation level.

Pelvis target

Range: Set of collision or mass objects. **Default value:** Empty.

Specifies a set of collision objects on the tackle target character which represent a pelvis target for the tackling character. The pelvis target must be specified in order to use a tackle behaviour. Usually a single collision object is sufficient to define the pelvis target. You can specify one or more mass objects as the pelvis target, although be aware that mass object positions are not updated when the tackle target character is in the Collision Only simulation level.

Writhe

Writhe is an active behaviour. The character will try to move its arms and legs in kicking or thrashing motion. The behaviour variables for this behaviour are:

VARIABLES

Amplitude

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative extent to which the arms and legs will swing. With minimum amplitude, the arms and legs will remain rigid. With maximum amplitude, the arms and legs will swing wildly.

Velocity

Range: -1.0 to 1.0. **Default value:** -0.6.

Specifies the relative rate at which the arms and legs will cycle during the writhe. With minimum speed, arms and legs will complete a cycle in around 500 frames. With maximum speed, arms and legs will cycle in around 20 frames.

Phase offset

Range: -1.0 to +1.0. **Default value:** 0.2.

Specifies the starting point for the oscillations that drive the arms and legs. Zero offset implies an ideal starting point. A maximum negative offset implies the arms and legs oscillate one cycle behind the ideal oscillation. A maximum positive offset implies the arms and legs oscillate one cycle ahead the ideal oscillation.

Strength

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative power applied to drive the arm and leg joints. With minimum strength, the arms and legs will fully cycle only if unobstructed and in zero gravity. With maximum strength, the arms and legs will fully cycle regardless of most obstacles.

Noise level

Range: 0.0 to 1.0. **Default value:** 0.0.

Specifies the relative amount of randomness applied to the behaviour. With minimum noise level, no noise is applied to the behaviour. With maximum noise level, maximum noise is applied. Adding randomness to a behaviour is a good way of generating more realistic animation.

Random number seed

Range: 1 to 10000. **Default value:** Random.

Establishes a sequence of random numbers used to add randomness, or noise, to the behaviour. Adding noise can add an additional level of realism to a behaviour. A particular random number seed will establish a repeatable noise sequence.

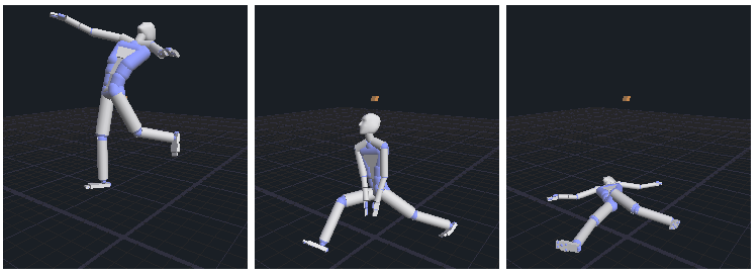
Open legs

Range: On/Off. **Default value:** Off.

Specifies the character will widen its legs separation during a writhe cycle.

Writhe In Mid-Air

Writhe In Mid-Air is a conditional behaviour. If the character is airborne, it will try to move its arms and legs in kicking or thrashing motion. This behaviour is conditional because it requires that the character is initially airborne. The behaviour variables for this behaviour are:



VARIABLES

Amplitude

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative extent to which the arms and legs will swing. With minimum amplitude, the arms and legs will remain rigid. With maximum amplitude, the arms and legs will swing wildly.

Speed

Range: 0.0 to 1.0. **Default value:** -0.6.

Specifies the relative rate at which the arms and legs will cycle during the writhe. With minimum speed, arms and legs will complete a cycle in around 500 frames. With maximum speed, arms and legs will cycle in around 20 frames.

Phase offset

Range: -1.0 to +1.0. **Default value:** 0.2.

Specifies the starting point for the oscillations that drive the arms and legs. Zero offset implies an ideal starting point. A maximum negative offset implies the arms and legs oscillate one cycle behind the ideal oscillation. A maximum positive offset implies the arms and legs oscillate one cycle ahead the ideal oscillation.

Strength

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative power applied to drive the arm and leg joints. With minimum strength, the arms and legs will fully cycle only if unobstructed and in zero gravity. With maximum strength, the arms and legs will fully cycle regardless of most obstacles.

Noise level

Range: 0.0 to 1.0. **Default value:** 0.0.

Specifies the relative amount of randomness applied to the behaviour. With minimum noise level, no noise is applied to the behaviour. With maximum noise level, maximum noise is applied. Adding randomness to a behaviour is a good way of generating more realistic animation.

Random number seed

Range: 1 to 10000. **Default value:** Random.

Establishes a sequence of random numbers used to add randomness, or noise, to the behaviour. Adding noise can add an additional level of realism to a behaviour. A particular random number seed will establish a repeatable noise sequence.

Open legs

Range: On/Off. **Default value:** Off.

Specifies the character will widen its legs separation during a writhe cycle.

Head turn

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative extent to which the character turns its head while airborne. With minimum head turn, the character will maintain its initial head angle. With maximum head turn, the head will fully turn in a direction that is determined by the trajectory of the character.

Back arch

Range: 0.0 to 1.0. **Default value:** 1.0.

Specifies the relative extent to which the character arches its back while airborne. With minimum back arch, the back will maintain its original joint angles. With maximum back arch, the back will arch as far as its joint angles allow.

Other behaviours

There are a number of behaviours that are designed to modify physical characteristic of the character, such as its **joint stiffness** and **body damping**. These are: Hold; Body Damping; Body Stiffness; and Whole Body Stiffness. These behaviours do not generate intelligent motion in themselves; rather they are used in conjunction with other behaviours to generate specific effects.

- **Body Damping** is an active behaviour. You can use this behaviour to change the linear or angular damping of the mass objects in the target mass object set.
- **Body Stiffness** is an active behaviour. You can use this behaviour to stiffen or relax various body parts of the character.
- **Hold** is an active behaviour. The hold behaviour is effectively a **dynamic constraint** which is enabled when two user-defined bodies collide or come within some user-defined distance from each other. The constraint then remains active for an adjustable period of time after which it is disabled. It will remain disabled for a further adjustable period after which it will again become active.
- **Whole Body Stiffness** is an active behaviour. You can use this behaviour to stiffen or relax the entire body of the character.

Body Stiffness

Body Stiffness is an active behaviour. You can use this behaviour to stiffen or relax various body parts of the character. The behaviour variables for this behaviour are:

VARIABLES

Head and neck stiffness

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative ability of the head and neck to maintain their initial pose during the behaviour event. With minimum stiffness, the head and neck will become limp. With maximum stiffness, the head and neck will remain rigid.

Torso stiffness

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative ability of the torso to maintain its initial pose during the behaviour event. With minimum stiffness, the torso will become limp. With maximum stiffness, the torso will remain rigid.

Left arm stiffness

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative ability of the left arm to maintain its initial pose during the behaviour event. With minimum stiffness, the left arm will become limp. With maximum stiffness, the left arm will remain rigid.

Right arm stiffness

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative ability of the right arm to maintain its initial pose during the behaviour event. With minimum stiffness, the right arm will become limp. With maximum stiffness, the right arm will remain rigid.

Left leg stiffness

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative ability of the left leg to maintain its initial pose during the behaviour event. With minimum stiffness, the left leg will become limp. With maximum stiffness, the left leg will remain rigid.

Right leg stiffness

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative ability of the right leg to maintain its initial pose during the behaviour event. With minimum stiffness, the right leg will become limp. With maximum stiffness, the right leg will remain rigid.

Body Damping

Body Damping is an active behaviour. The can use this behaviour to change the linear or angular damping of the mass objects in the target mass object set. The behaviour variables for this behaviour are:

VARIABLES: Angular Body Damping

[Joint]

Range: 0.0 to 100.0. **Default value:** 10.0.

Specifies the extent to which damping is applied to rotations of the joint. With minimum damping, the joint rotates freely. With maximum damping, the joint rotates very slowly.

There are separate angular body damping coefficients for the following joints: Neck, UpperSpine, MiddleSpine, LowerSpine, Pelvis, LeftShoulder, LeftUpperArm, LeftForearm, LeftHand, RightShoulder, RightUpperArm, RightForearm, RightHand, LeftUpperLeg, LeftLowerLeg, LeftFoot, RightUpperLeg, RightLowerLeg and RightFoot.

VARIABLES: Linear Body Damping

[Joint]

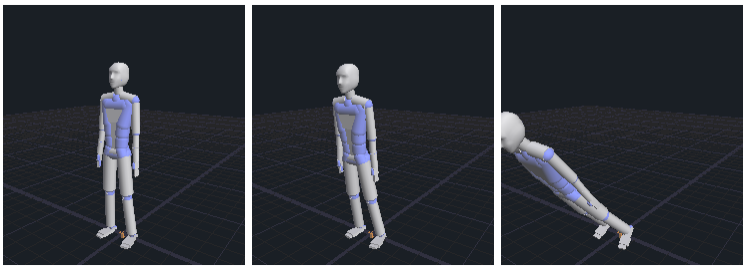
Range: 0.0 to 100.0. **Default value:** 10.0.

Specifies the extent to which damping is applied to movements of the joint. With minimum damping, the joint moves freely. With maximum damping, the joint moves very slowly.

There are separate linear body damping coefficients for the following joints: Neck, UpperSpine, MiddleSpine, LowerSpine, Pelvis, LeftShoulder, LeftUpperArm, LeftForearm, LeftHand, RightShoulder, RightUpperArm, RightForearm, RightHand, LeftUpperLeg, LeftLowerLeg, LeftFoot, RightUpperLeg, RightLowerLeg and RightFoot.

Whole Body Stiffness

Whole Body Stiffness is an active behaviour. You can use this behaviour to stiffen or relax the entire body of the character. The behaviour variables for this behaviour are:



VARIABLES

Stiffness

Range: 0.0 to 1.0. **Default value:** 0.6.

Specifies the relative ability of the whole body to maintain its initial pose during the behaviour event. With minimum stiffness, the whole body will become limp. With maximum stiffness, the whole body will remain rigid.

Hold

Hold is an active behaviour. The hold behaviour is effectively a **dynamic constraint** which is enabled when two user-defined bodies collide or come within some user-defined distance from each other. The constraint then remains active for an adjustable period of time after which it is disabled. It will remain disabled for a further adjustable period after which it will again become active.

This behaviour can be used to allow characters to catch objects in motion, or to hold onto static objects when the character is in motion. One advantage of this dynamic constraint is that it is not necessary to know in advance exactly when the constraint should be activated. The behaviour variables for this behaviour are:

VARIABLES

Hold Duration

Range: 0.0 to 1000.0. **Default value:** 2.0.

Time in seconds that the ConstraintFromObject will be held to the ConstraintToObject.

Disable Duration

Range: 0.0 to 1000.0. **Default value:** 2.0.

Time in seconds hold will be inactive. After this time it will go back into standard hold mode.

Use Proximity Trigger

Range: On/Off. **Default value:** On.

Allows the two selected objects to trigger on a user specified distance.

Proximity

Range: >0 to 1000.0. **Default value:** 0.2.

Threshold distance between the two selected objects. Once below, the hold constraint will trigger.

ConstraintFromObject

Range: Set of mass objects. **Default value:** Empty.

Mass object that will hold to the other mass object.

ConstraintToObject

Range: Set of mass objects. **Default value:** Empty.

Mass object that other mass object will hold to.

Constraint Type

Range: Lock/Lock Position/Lock Orientation/Join Bodies. **Default value:** Join Bodies.

Type of constraint that is to be created between the associated mass objects.

- **Lock** will prevent the target mass objects from moving or rotating.
- **Lock Position** will prevent the target mass objects from moving, but allows the objects to rotate.
- **Lock Orientation** will prevent the target mass objects from rotating, but allows the objects to move.
- **Join Bodies** forces the movement and rotation of the target mass objects to be locked together. That is, the mass objects can collectively move and rotate as a single rigid body. They cannot move or rotate with respect to each other.

  Use a Join Bodies constraint type when you want to model a connection between two mass objects in which each object can affect the other objects. This is most often used when all the mass objects are being driven by the dynamic simulation. Join Bodies allows the momentum and linear momentum of each object to affect the other objects in a **two-way** direction.

Position Strength X

Range: Unlimited positive value. **Default value:** 100000.0.

Positional strength of the constraint in the X axis

Does not apply to the Lock Orientation constraint type. Strengths are dimensionless values that are converted into constraint forces internally. Use very large constraint strengths (in the range 1000000.0) to **rigidly** constrain movement in a given direction. Use small constraint strengths (in the range 1000.0) to provide a **soft** movement constraint in a given direction. Use zero constraint strengths to allow **free** movement in a given direction.

Position Strength Y

Range: Unlimited positive value. **Default value:** 100000.0.

Positional strength of the constraint in the Y axis.

Position Strength Z

Range: Unlimited positive value. **Default value:** 100000.0.

Positional strength of the constraint in the Z axis.

Angular Strength X

Range: Unlimited positive value. **Default value:** 100000.0.

Orientation strength of the constraint in the X axis.

Does not apply to the Lock Position constraint type. Strengths are dimensionless values that are converted into constraint forces internally. Use very large constraint strengths (in the range 1000000.0) to **rigidly** constrain rotation about a given direction. Use small constraint strengths (in the range 1000.0) to provide a **soft** rotation constraint about a given direction. Use zero constraint strengths to allow **free** rotation in a given direction.

Angular Strength Y

Range: Unlimited positive value. **Default value:** 100000.0.

Orientation strength of the constraint in the Y axis.

Angular Strength Z

Range: Unlimited positive value. **Default value:** 100000.0.

Orientation strength of the constraint in the Z axis.

Chapter 19

Character Edit Mode

What are *endorphin* custom characters?

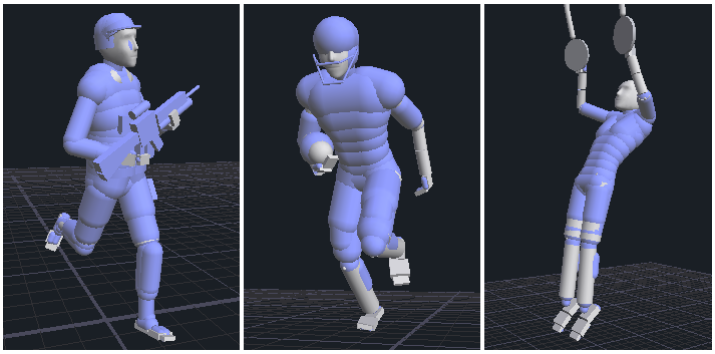
The *endorphin* standard simulation character represents an average-sized adult male human. However, you can also define **custom characters** that have different sizes and shapes to this standard character.

There are three main categories of custom characters: **simulation** characters, **reference** characters and **prop** characters.

Simulation and **reference** characters are frequently edited together to form a **pair** of matching characters. You will usually use reference characters in conjunction with simulation characters to model humans, humanoids and other bipedal characters. In contrast, you will usually use **prop** characters to model weapons, vehicles and other rigid, inanimate objects.

A key distinction between simulation characters, reference characters and prop characters is that you can apply adaptive **behaviour** events to simulation characters. You cannot apply adaptive behaviour events to reference characters or prop characters.

Simulation characters are saved as **.nmc** binary files. Reference characters and prop characters are saved as **_ref.nmc** binary files.



Simulation characters

You can modify the standard simulation character in many ways. For example, you can edit its joint lengths if you want your character to be taller or shorter. This is often called **resizing** the character.

You can edit its mass objects and collision objects if you want your character to be lighter or heavier. This is often called **reshaping** the character. You can also add some additional collision objects if you want to add definition to the character, or to add props such as armour or weapons.

To create custom simulation characters, you modify the **standard simulation character** and then save it as a custom simulation character. You can populate a scene with instance of many different custom characters. You can apply behaviour events to your custom characters—the adaptive nature of behaviours means they work just as well on a range of character shapes and sizes.

You are not limited to creating humanlike custom characters. As long as your custom character is essentially bipedal, your custom character could represent any kind of fantasy character, from orcs and dwarfs to ogres and giant apes.

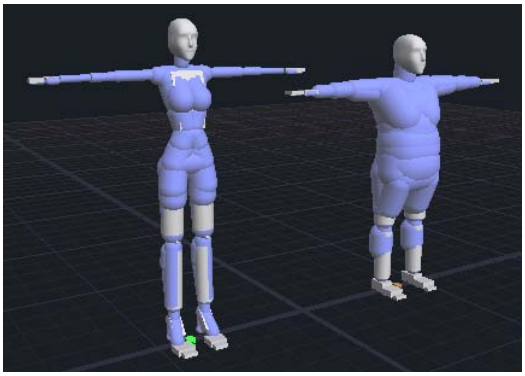
Custom simulation characters are saved as **.nmc** binary files.

Standard simulation character

endorphin ships with a standard simulation character, **Standard Simulation Character.nmc**, which is used as the basis for new custom simulation characters. The standard simulation character is stored in **Resources\Characters\Standard**. Do not move, edit or delete this file.

Sample simulation characters

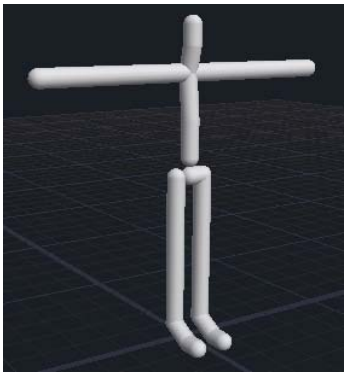
endorphin ships with a number of sample simulation characters. Browse to the **Resources\Characters\Custom** folder for examples. Many of the *endorphin* sample scenes also include custom simulation characters such as football players.



Reference characters

Most of the time when you are using *endorphin* as part of a pipeline, you will be using it in conjunction with other character hierarchies. These hierarchies may have different numbers of joints, and with different naming conventions, to the *endorphin* simulation character. You can create *endorphin* characters using your own joint hierarchies. These are called **custom reference characters**. Custom reference characters are created in **Character Edit Mode** by importing an existing character joint hierarchy contained in an FBX, XSI, BVH, ASF or VST/VSK file.

Custom reference characters are saved as **_ref.nmc** binary files.



Using custom reference characters

Most of the time, you do not actually populate an *endorphin* virtual world with custom reference characters. Rather you populate the virtual world with corresponding custom simulation characters that have been **reshaped** and **resized** to match your custom reference characters.

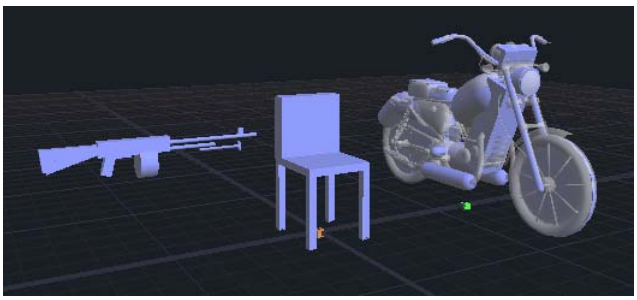
Typically, you will animate in *endorphin* using custom simulation characters, and then **map** this motion onto the corresponding custom reference character using a process called **dynamic motion transfer**. You will then export the motion of the custom reference character into an external FBX, XSI, BVH, AMC, CSM or V file. You can also use dynamic motion transfer in the other direction. That is, you can import your own animation onto a custom reference character, and then use dynamic motion transfer to map the motion onto an equivalent custom simulation character.

Prop characters

You can also create **custom prop characters**.

Prop characters are used to represent all kinds of animate and inanimate objects in the virtual world. You can create simple, rigid prop characters that represent weapons, vehicles, furniture and other scene objects. You can also create more complex, articulated structures that represent creatures such as horses or dinosaurs.

Custom prop characters are saved as **_ref.nmc** binary files.



Prop characters and reference characters

Prop characters are a type of **reference** character. The key distinction between simulation characters, reference characters and prop characters is that you can apply adaptive behaviour events to simulation characters. You cannot apply adaptive behaviour events to reference characters or prop characters.

The distinction between reference characters and prop characters is that you use reference characters in conjunction with simulation characters for **motion data transfer** in your animation pipeline. In contrast, you use prop characters to represent weapons, vehicles, creatures and other objects in the virtual world during dynamic simulations.

Using the standard prop character

You create custom prop characters in Character Edit Mode. If you are creating a single, rigid object, you might want to use the **standard prop character** as the basis for the character. The standard prop character is made up of a single joint, a single mass object and a single collision object. You can always add more collision objects to the character.

The standard prop character is **Standard Prop Character.nmc** and is shipped in the **Resources\Characters\Standard** folder. Do not move, edit or delete this file.

Creating a prop character from external file

You can create custom prop characters be loading in a skeletal file from an external package. This can have any number of joints but usually will have a single joint for objects

like chairs, guns and so on. Objects with articulated parts, a helicopter for example would have multiple joints.

Sample prop characters

endorphin ships with a number of sample prop characters. Browse to the **Resources\Props** folder for examples of different types of vehicle, weapon and furniture props.

- **Vehicles** include Porsche_ref.nmc, Motorcycle_ref.nmc, Lorry_ref.nmc, Biplane_ref.nmc and Helicopter_ref.nmc.
- **Weapons** include Gun Small_ref.nmc and Gun Large_ref.nmc.
- **Furnishings** include Chair_ref.nmc, RoundTable_ref.nmc and BreakingTable_ref.nmc.

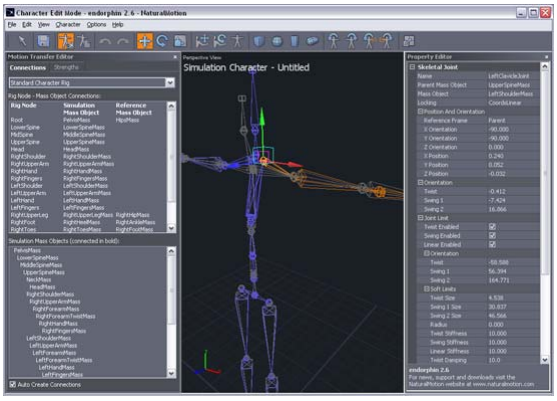
Third-party scripts

endorphin ships with various scripts to allow you to create prop characters from these systems and export into *endorphin*. These are plain text files with the extension **.nmc**.

- The Maya MEL script **endorphin_Exporter.mel** can be run in Maya to save Maya animated cameras as *endorphin* **.nmc** camera files. This file is shipped in the **Resources\Third Party\Scripts\Maya** folder. Alongside the MEL script file is an accompanying text file, **endorphin_Exporter_ReadMe.txt**, that includes more detailed help.
- The 3dsMax MAXscript **endorphin_CameraExporter.ms** can be run in 3dsMax to save 3dsMax animated cameras as *endorphin* **.nmc** camera files. This file is shipped in the **Resources\Third Party\Scripts\3dsMax** folder. Alongside the script file is an accompanying text file, **endorphin_MaxScripts.txt**, that includes more detailed help.

Introducing Character Edit Mode

endorphin has a special editing mode for creating, editing and saving custom characters called **Character Edit Mode**.



Working in Character Edit Mode

Many of the tools that are available for editing scenes, such as the Move, Rotate, Scale, Pose Move and Pose Rotate tools, are also available in Character Edit Mode. There are also some additional tools, such as the Mirror tools, that are only available in Character Edit Mode.

You can use the viewports, Property Editor and Node View to edit characters in Character Edit Mode. However, there is no Timeline Editor available, as there is no concept of running a simulation in Character Edit Mode.

Creating, editing and saving characters

You can use Character Edit Mode to create, edit and save the following types of characters:

- Custom **simulation** characters based on the **Standard Simulation Character** file.
- Custom **reference** characters based on **external** character hierarchy definitions stored in FBX, XSI, BVH, ASF or VST/VSK files.
- Custom **prop** characters based on **external** character hierarchy definitions stored in FBX, XSI, BVH, ASF or VST/VSK files, or based on the **Standard Prop Character** file.

Editing custom characters

Regardless of whether you are editing a custom **simulation**, **reference** or **prop** character in **Character Edit Mode**, you can perform the same reshaping and resizing operations to all types of character. The available operations are:

Editing joints

You can change the length of any joints in a custom character. For example, you could make most joints longer if you want to create a tall character, or make most joints shorter if you want to create a short character. You could make the arms longer and legs shorter for a primate-style character. Joint lengths do not have to be the same on the left and right sides of the character.

You can edit simulation character joints, but you **cannot** change the joint hierarchy itself, and you cannot add or remove any joints.

Editing joint limits

You can change any of the joint hard or soft limits, and other properties of joint limits such as damping and stiffness, in a custom character. You can also change the default position of each joint. For example, if you wanted to create a hunchback character, you might reduce the joint limits for the spine joints and change their default positions so that the spine joints hunch over. Joint limits do not have to be the same on the left and right sides of the character.

Editing mass objects

You can change the size, shape and density of any of the mass objects in a custom character. For example, if you want to create a heavier character you could increase the size or density of some of the mass objects. If the character is carrying a heavy item, you might reflect this by increasing the size or density of a character mass object that corresponds to the location of the item. Mass objects do not have to be the same on the left and right sides of the character.

You can edit simulation character mass objects, but you cannot add or remove any mass objects.

Editing, adding and deleting collision objects

You can change the **size**, **shape** and **material** of any of the collision objects in a custom character. You can also **add** and **remove** collision objects.

For example, if you want to create a large, rotund character you could add some additional collision objects to the pelvis or torso to reflect this additional size. Alternatively, if your character is wearing a helmet you could add some additional collision objects to the head to reflect this. Collision objects do not have to be the same

on the left and right sides of the character. When you create new collision objects, you must specify which mass object they are associated with.

Editing graphical objects

The standard *endorphin* character has a **HeadGraphic** graphical object associated with the **HeadMass** mass object. The HeadGraphic object helps make the orientation of the *endorphin* character more obvious.

Simulation characters also have small graphical objects that indicate the position of the left and right **elbows** and **knees**. These objects are useful when examining arm and leg rotations. These are the **LeftElbowGraphic**, **RightElbowGraphic**, **LeftKneeGraphic** and **RightKneeGraphic** objects respectively.

In the current version of *endorphin*, these helper graphical objects are **always** added to characters in scenes, even if you explicitly delete these objects from the corresponding character definitions in Character Edit Mode. This behaviour may change in future releases of *endorphin*.

If you do not want to see these graphical objects, you can **hide** them from the scene using the **Node View**. Right-click on the graphic objects that you want to hide, and then right-click and select **Hide** from the context menu. These visibility settings are stored in the scene file and are used when the scene file is reopened later.

Character Edit Mode grid

When you are editing a character in Character Edit Mode, *endorphin* displays the **Character Edit Mode grid**, rather than the usual **scene grid**. By default, the Character Edit Mode grid is **smaller** than the scene grid. Keep in mind that if you modify the grid size or density while in Character Edit Mode, you affect the Character Edit Mode grid, rather than the scene grid.

Posing and testing characters

When you are creating a new custom character, you generally use the Move, Rotate, Scale and Mirror tools to **reshape** the character joint hierarchy by editing its joints, joint limits, mass objects and collision objects.

It is generally good practice to avoid using the Pose Move and Pose Rotate tools while you are reshaping a character, as it can be confusing as to which operation are actually **modifying** a character definition, and which operations are **exploring** a character definition.

Common problems in new characters

A common problem when creating new simulation characters is that collision objects inadvertently interpenetrate without the appropriate Collides With Parent/Siblings/Ancestors setting turned off. Another common problem is that joint limit indicators end up outside the allowable joint limit range. If you attempt to simulate a character with either of these problems, the character will not work correctly when simulated. There will be a noticeable **popping** during the first few frames of a simulation in which the offending collision objects or joints rapidly self-correct.

Using pose tools to test characters

Pose tools can be useful when you want to test a character design. When you use the Pose Move or Pose Rotate tools with a custom character in Character Edit Mode, *endorphin* internally **simulates** the character when you begin dragging using either of these tools. Any collision object or joint limit problems will become apparent by the same **popping** behaviour as the character attempts to self-correct.

In addition to identifying obvious errors, posing is also a good way to experiment with a character design. You can assess the appropriateness of joint limits, for example, by posing the character in various ways.

To reset the character pose:

It is critical that after testing the character by posing it, you **reset** the character pose back to the default pose. If you do not reset the pose, subsequent editing can become very confusing, as the character will be in a mix of its default and posed states.

Pose resetting generally occurs **automatically** whenever you exit the Pose Move or Pose Rotate tools—such as when activating the Move, Rotate or Scale tools. The pose is also reset automatically if you attempt to **save**, **activate**, **show** or **hide** a character, or when running one of the **Mirror** tool commands.

Prior to *endorphin* 2.6, you had to reset character poses **manually**. To manually reset a character pose:

1. Ensure the character, or some part of the character, is selected.
2. Click the **Reset Pose** button  in the Main Toolbar to reset the pose. There is no keyboard shortcut for this command.

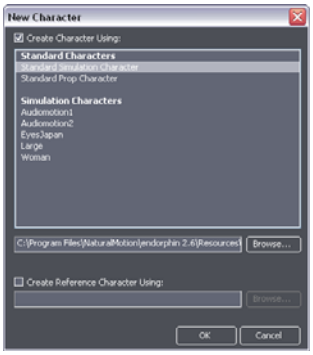
Using the Mirror tool while posing

You should generally avoid using the Mirror tools when using the pose tools in Character Edit Mode. If you **do** want to use the Mirror tools to mirror the positions of posed limbs, you should take care to ensure that mirroring is only applied to arms and legs, and not to other body parts which cross the Y-Axis centreline of the character. This can be achieved when the posing tools are active by **freezing** the central body parts. You can freeze mass

objects by right-clicking them in the viewports. Frozen mass objects are displayed using the grey **frozen mass object** colour. Including central body parts in Mirror tool selection sets when the pose tools are active will generate unpredictable (and usually wrong) results.

Creating new simulation characters

You can create new custom **simulation** characters by using the standard simulation character, or an existing custom simulation character, as the basis for the new simulation character.



To create a new simulation character:

- Select **File > New Character…** This displays the **New Character** dialog. There is no equivalent keyboard shortcut.
- Turn on **Create Character Using** and turn off **Create Reference Character Using**.
- The list of simulation characters will **always** include the **Standard Simulation Character**. This is the standard *endorphin* character, and represents an average-sized adult human male.

  The list may also contain a list of **Simulation Characters**. These are custom simulation characters that have been derived from the standard simulation character. Custom simulation characters have the same joint hierarchy as the standard simulation character, but can have a different shape and size.
- The **Path** text box contains the location of your custom character folder. Click the **Browse…** button to display the **Browse For Folder** dialog. You can browse to a different character folder. The list of Simulation Characters updates to reflect the characters that the system finds in the character folder that you have specified.

  The Path character folder is remembered by *endorphin*. This means that once you have browsed to a character folder once, you do not have to keep on browsing to

the same location. Regardless of the character folder you have selected, the standard simulation character is always available in the character list.

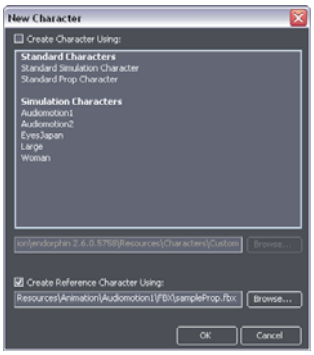
- Select a character that you want to use as the basis for your new custom simulation character. You can select the standard simulation character or any of the existing custom simulation characters.
- Click **OK** to create a new custom simulation character based on the selected character. Alternatively, **double-click** the selected character.
- The dialog closes and *endorphin* enters **Character Edit Mode**. The new character is displayed at the centre of the virtual world in its default T-pose. The virtual world is empty except for this new character. The viewport camera is reset.

  In Character Edit Mode, the Main Toolbar has a different set of buttons, and the Menu Bar also has a different set of commands available. In addition, the Timeline Editor is not unavailable while you are in Character Edit Mode.

  When you create a new simulation character, the label **Simulation Character - Untitled** will appear in the viewports until you save the character.

Creating new prop characters

You can create new custom **prop** characters by importing a node or joint hierarchy stored in an external animation file in any of the FBX, XSI, BVH, ASF or VST/VSK file formats. You can also create new custom prop characters using the **standard prop character**, or an existing custom prop character, as the basis for the new prop character.



To create a new prop character using an external file:

1. Select **File > New Character…** This displays the **New Character** dialog. There is no equivalent keyboard shortcut.
2. Turn on **Create Reference Character Using** and turn off **Create Character Using**.

3. The **Reference Character Path** text box contains the location of the external file that defines your reference character joint hierarchy. Click the **Browse…** button to display the **Select Reference Character File** dialog.

4. Select an external file that contains your reference character joint hierarchy. An external file should be chosen that contains your reference character defined in a **standard default pose**, such as a T-pose. This will make dynamic motion transfer more reliable. Reference data can be stored in FBX, XSI, BVH, ASF or VST/VSK files.

5. Click **OK**. The dialog closes and *endorphin* displays the **Import Options** dialog. This dialog allows you to specify many aspects of the import process. You can specify the units, angular units, scale, coordinate system and orientation of your source reference file. You can also specify which node in the reference character file should act as the **root joint** of the corresponding reference character.

   For more information on using the Import Options dialog, see the **Import Options** topic.

6. Click **OK**. The dialog closes and *endorphin* enters **Character Edit Mode**. The new character is displayed at the centre of the virtual world in its default pose. The virtual world is empty except for this new character. The viewport camera is reset.

   In Character Edit Mode, the Main Toolbar has a different set of buttons, and the Menu Bar also has a different set of commands available. In addition, the Timeline Editor is not unavailable while you are in Character Edit Mode.

   When you create a new reference character, the label **Reference Character - Untitled** will appear in the viewports until you save the character.

To create a new prop character using the Standard Prop Character:

1. Select **File > New Character…** This displays the **New Character** dialog. There is no equivalent keyboard shortcut.

2. Turn on **Create Character Using** and turn off **Create Reference Character Using**.

3. The list of characters will **always** include the **Standard Prop Character**. This is the standard *endorphin* prop character. The standard prop character contains a single mass object and a single collision object.

   The list may also contain a list of **Prop Characters**. These are custom prop characters that have been derived from the standard prop character. Custom prop characters have the same joint and mass object found in the standard prop character, but may also have additional collision objects.

4. The **Path** text box contains the location of your custom character folder. Click the **Browse…** button to display the **Browse For Folder** dialog. You can browse to a different character folder. The list of Simulation Characters updates to reflect the characters that the system finds in the character folder that you have specified.

The Path character folder is remembered by *endorphin*. This means that once you have browsed to a character folder once, you do not have to keep on browsing to the same location. Regardless of the character folder you have selected, the standard simulation character is always available in the character list.

5. Select a character that you want to use as the basis for your new custom prop character. You can select the standard prop character or any of the existing custom prop characters.
6. Click **OK** to create a new custom prop character based on the selected prop character. Alternatively, **double-click** the selected prop character.
7. The dialog closes and *endorphin* enters **Character Edit Mode**. The new character is displayed at the centre of the virtual world. The virtual world is empty except for this new character. The viewport camera is reset.

   In Character Edit Mode, the Main Toolbar has a different set of buttons, and the Menu Bar also has a different set of commands available. In addition, the Timeline Editor is not unavailable while you are in Character Edit Mode.

   When you create a new prop character, the label **Reference Character - Untitled** will appear in the viewports until you save the character.

   All prop characters contain a root joint called **root**, a single mass object called **rootMass** and a single collision object called **RootCollision**. You can edit all of these objects. You can also create additional collision objects to build up a more complex collision surface.

Creating a new simulation-reference character pair

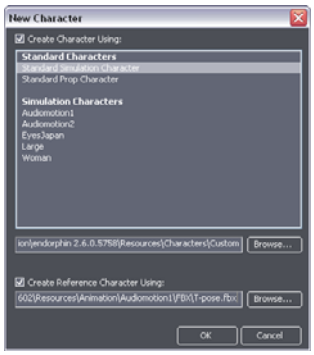
You will often create simulation and reference characters together as a single **pair** of characters.

Typically, when you are working with *endorphin* as part of an **animation pipeline**, you will have your character node hierarchy stored in an **external** file, and you will be using *endorphin* to generate animation that you want to use in conjunction with your external character definition.

The basic workflow is to create a new **simulation-reference character pair**, where the simulation character is derived from the standard simulation character, and the reference character is based on the external reference character definition. You will then **reshape** and **resize** the simulation character to match its corresponding reference character.

This will allow you to simulate scenes in *endorphin* using the simulation character as a **stand-in** for its reference character. You will be able to use **dynamic motion transfer** to **import** motion from your reference character onto your simulation character. After

generating motion in *endorphin* using the simulation character, you will be able to **export** motion back out from your simulation character to your reference character.



To create a new simulation-reference pair:

1. Select **File > New Character…** This displays the **New Character** dialog. There is no equivalent keyboard shortcut.
2. Turn on **Create Character Using**.
3. The list of simulation characters will **always** include the **Standard Simulation Character**. This is the standard *endorphin* character, and represents an average-sized adult human male.

   The list may also contain a list of **Simulation Characters**. These are custom simulation characters that have been derived from the **standard simulation character**. Custom simulation characters have the same joint hierarchy as the standard simulation character, but can have a different shape and size.
4. The **Path** text box contains the location of your custom character folder. Click the **Browse…** button to display the **Browse For Folder** dialog. You can browse to a different character folder. The list of Simulation Characters updates to reflect the characters that the system finds in the character folder that you have specified.

   The Path character folder is remembered by *endorphin*. This means that once you have browsed to a character folder once, you do not have to keep on browsing to the same location. Regardless of the character folder you have selected, the standard simulation character is always available in the character list.
5. Select a character that you want to use as the basis for your new custom simulation character. You can select the standard simulation character or any of the existing custom simulation characters.
6. Turn on **Create Reference Character Using**.
7. The **Reference Character Path** text box contains the location of the external file that defines your reference character joint hierarchy. Click the **Browse…** button to display the **Select Reference Character File** dialog.

8. Select a reference file that contains your reference character joint hierarchy. A reference file should be chosen that contains your reference character defined in a **standard default pose**—such as a T-pose—and facing forward in the **Z axis**. This will make dynamic motion transfer more reliable. Reference data can be stored in FBX, XSI, BVH, ASF or VST/VSK files.

9. Click **OK**. The dialog closes and *endorphin* displays the **Import Options** dialog. This dialog allows you to specify many aspects of the import process. You can specify the units, angular units, scale, coordinate system and orientation of your source reference file. You can also specify which node in the reference character file should act as the **root joint** of the corresponding reference character.

   For more information on using the Import Options dialog, see the **Import Options** topic.

10. Click **OK**. The dialog closes and *endorphin* enters **Character Edit Mode**.

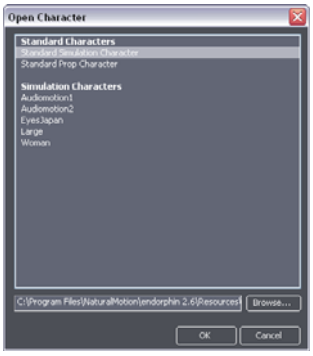
11. There are now **two** new characters displayed at the centre of the virtual world, a simulation character and its reference character. The simulation character will be in its default pose. The reference character will be in the pose stored in the external reference character file. The virtual world is empty except for these new characters. The viewport camera is reset.

    In Character Edit Mode, the Main Toolbar has a different set of buttons, and the Menu Bar also has a different set of commands available. In addition, the Timeline Editor is not unavailable while you are in Character Edit Mode.

    When you create a new simulation-reference character pair, the label **Simulation Character - Untitled** will appear in the viewports until you save the character.

Opening existing characters

You can open existing **simulation**, **reference** and **prop** characters. You can also open **simulation-reference character pairs** at the same time.



To open an existing character:

1. Select **File > Open Character…**. There is no keyboard shortcut for this command.
2. The **Open Character** dialog will be displayed.
3. The **Standard Characters** section will always contain the standard simulation character and the standard prop character.
4. The list may also contain a list of **Simulation Characters** and it may also contain a list of **Prop Characters**. The Simulation Characters list will contain any characters with a **.nmc** extension in the character folder. The Prop Characters list will contain any characters with a **_ref.nmc** extension for which there is no corresponding simulation character of the same name.
5. The **Path** text box contains the location of your custom character folder. Click the **Browse…** button to display the **Browse For Folder** dialog. You can browse to a different character folder. The lists of Simulation Characters and Prop Characters update to reflect the characters that the system finds in the character folder that you have specified.

   The Path character folder is remembered by *endorphin*. This means that once you have browsed to a character folder once, you do not have to keep on browsing to the same location. Regardless of the character folder you have selected, the standard simulation character is always available in the character list.
6. Click **OK** to open the selected character. Alternatively, **double-click** the selected prop character.

   If you have selected a custom simulation character, and there is a corresponding reference character of the same name in the same folder, that reference character will also be opened.

- Suppose your character folder contains a pair of simulation and reference characters with the filenames **Goblin.nmc** and **Goblin_ref.nmc**. The Open Character dialog will display **Goblin** as a character in the Simulation Characters section. Selecting **Goblin** will open both **Goblin.nmc** and **Goblin_ref.nmc** into Character Edit Mode.
- Suppose your character folder contains a prop character with the filename **Rifle_ref.nmc**. It is assumed to be a prop character, rather than a reference character, because there is no equivalent simulation character file. The Open Character dialog will display **Rifle** as a character in the **Prop Characters** section. Selecting **Rifle** will open **Rifle_ref.nmc** into Character Edit Mode.

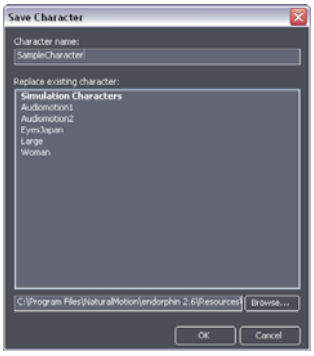
Learning Edition characters

Keep in mind that characters created in *endorphin* Learning Edition cannot be opened in *endorphin* or *endorphin* Educational Edition, and vice versa.

Saving characters

You can save **simulation**, **reference** and **prop** characters as new characters, or to replace existing characters.

You will often be working **simulation-reference character pairs** in Character Edit Mode. In this case, you actually save out to **two** separate files. The simulation character will be named after the character, with a **.nmc** extension. The reference character will also be named after the character, but with a **_ref.nmc** extension.



To save the character as a new file:

1. Select **File > Save Character**. The keyboard shortcut is **Ctrl+S**. Alternatively, you can click the **Save Character** button  in the Main Toolbar.
2. If you have not already saved the character or characters, the **Save Character** dialog will be displayed.
3. The **Path** text box contains the location of your custom character folder. Click the **[Browse]** button to display the **Browse For Folder** dialog. You can browse to a different character folder. The list of Simulation Characters or Prop Characters updates to reflect the characters that the system finds in the character folder that you have specified.

   The Path character folder is remembered by *endorphin*. This means that once you have browsed to a character folder once, you do not have to keep on browsing to the same location. Regardless of the character folder you have selected, the standard simulation character is always available in the character list.
4. Enter the name of your character into the **Character Name** text box. You can use spaces in the character name. For example, **Goblin** and **Ice Monster** are both valid character names.
5. Click **OK** to save the characters in Character Edit Mode to the folder specified by the **Path**.
   - If you are editing a simulation character, a single **.nmc** file will be created, such as **Goblin.nmc**.
   - If you are editing a reference character, a single **_ref.nmc** file will be created, such as **Goblin_ref.nmc**.
   - If you are editing a simulation-reference character pair, two files will be created: a **.nmc** file and a **_ref.nmc** file, such as **Goblin.nmc** and **Goblin_ref.nmc**.
   - If you are editing a prop character, a single **_ref.nmc** file will be created, such as **Rifle_ref.nmc**.
6. When you save a character, the viewport label will update to reflect the name of the saved character. If the character had not bee previously saved, the label **Untitled** will be replaced with the saved character name.

To save the character over an existing file:

1. Select **File > Save Character**. The keyboard shortcut is **Ctrl+S**. Alternatively, you can click the **Save Character** button  in the Main Toolbar.
2. If you have not already saved the character or characters, the **Save Character** dialog will be displayed.

3. The **Path** text box contains the location of your custom character folder. Click the **Browse…** button to display the **Browse For Folder** dialog. You can browse to a different character folder. The list of Simulation Characters or Prop Characters updates to reflect the characters that the system finds in the character folder that you have specified.

   The Path character folder is remembered by *endorphin*. This means that once you have browsed to a character folder once, you do not have to keep on browsing to the same location. Regardless of the character folder you have selected, the standard simulation character is always available in the character list.

4. **Select** one of the Simulation or Prop character names. At this point it does not matter whether you are editing a simulation, reference or prop character. Selecting an existing character name simply specifies a character name to use.

5. Click **OK** to save the characters in Character Edit Mode. Alternatively, double-click on the Simulation or Prop character name to save the characters. Usually, this will mean replacing existing files. In some cases it may mean creating new files.

   When you save a character, the viewport label will update to reflect the name of the saved character.

   For example, you may be editing a simulation-reference character pair, and you may choose to save over an existing simulation character for which there is no corresponding reference character. In this case, the simulation character will be replaced and a new reference character will be created.

   - If you are editing a simulation character, a single **.nmc** file will be replaced or created, such as **Goblin.nmc**.
   - If you are editing a simulation-reference character pair, **two** files will be replaced or created: a **.nmc** file and a **_ref.nmc** file, such as **Goblin.nmc** and **Goblin_ref.nmc**.
   - If you are editing a prop character, a single **_ref.nmc** file will be replaced or created, such as **Rifle_ref.nmc**.

6. When you save a character to a new file, the viewport label will update to reflect the new name of the saved character.

Closing characters

There is no specific command to close a character. *endorphin* is a **single-document interface** system, which means that you can edit one character or scene at a time.

If you create a new scene or character, or open another existing scene or character, or close the application itself, then the characters currently being edited is closed automatically. If you have made any unsaved changes to the character, a **Save** dialog appears. This dialog gives you the opportunity of saving any changes you have made to the character.

Selecting characters

When you are editing a scene, the easiest way to select an entire character is to select it using the Timeline Editor. When you are editing characters in **Character Edit Mode**, the Timeline Editor is not available. However, there are a number of ways to select an entire character in Character Edit Mode.

Simulation characters

If you are editing a simulation character or a simulation-reference character pair, you can select the **simulation** character by:

- Clicking the **Activate Simulation Character** button  in the Main Toolbar;
- Right-clicking and selecting **Select Character > Simulation Character**;
- Selecting the **Simulation Character** node in the Node View.

Reference characters

If you are editing a reference character or a simulation-reference character pair, you can select the **reference** character by:

- Clicking the **Activate Reference Character** button  in the Main Toolbar.
- Right-clicking and selecting **Select Character > Reference Character**.
- Selecting the **Reference Character** node in the Node View.

Activating characters

When you are editing a **simulation-reference character pair**, the simulation and reference characters are both displayed in the viewport. However, only **one** of the pair of characters is **active** at any given time. The character that is not editable is **frozen**. Frozen characters cannot be selected in the viewport, although they can be selected in the Node View.

- If the **simulation** character is active, the viewport label will be **Simulation Character**.
- If the **reference** character is active, the viewport label will be **Reference Character**.

You cannot readily distinguish the active and frozen characters in the viewport when the viewport is in Shaded View. However, when the viewport is in **Skeletal View**, you can distinguish the characters. The active character joints are displayed in the blue joint colour, whereas the frozen character joints are displayed in the gray frozen joint colour.

To activate the simulation character:

- Click the **Activate Simulation Character** button in the Main Toolbar to set the simulation character active and the reference character frozen. This command also **selects** the simulation character.

- Alternatively, right-click and select **Select Character > Simulation Character**. This will select the simulation character, and will also activate the simulation character if it is not already active.

To activate the reference character:

- Click the **Activate Reference Character** button in the Main Toolbar to set the reference character active and the simulation character frozen. This command also **selects** the reference character.

- Alternatively, right-click and select **Select Character > Reference Character**. This will select the reference character, and will also activate the reference character if it is not already active.

To toggle the active character:

- Select **Edit > Toggle Active Character** to toggle the active character. The keyboard shortcut is **Ctrl+Tab**. This is a useful way to rapidly toggle between editing the simulation and reference characters. Note that this command is only available when you are editing a simulation-reference character pair.

Editing the local coordinate system

endorphin characters have a **local coordinate system**. The origin of the local character coordinate system is identified by the character cube in the viewport. The standard simulation character local origin is located between its feet when it is in the T-pose.

When you are creating new simulation characters, you can choose to redefine the position and orientation of the local character coordinate system. If you are planning on doing this, you should carry out this step **before** you begin any detailed reshaping and resizing of your simulation character.

For example, your animation pipeline might require that you redefine the character local coordinate system so that the character cube is at the same position as the character root joint. You may also need to redefine the orientation of the character coordinate system if necessary. This is a relatively advanced topic that can get quite complicated, and depends upon the details of your animation pipeline. Contact our **technical support team** for advice if your are having problems setting up your character coordinate system.

To edit the position or orientation of the local coordinate system:

1. **Activate** the character that you want to edit in Character Edit Mode. Usually this will be the simulation character.
2. Select the **character cube** in the viewport. If you cannot see the character cube, you may have to toggle the display of character cubes by selecting **View > Character Cubes**, or by right-clicking and selecting **Display > Character Cubes**.
3. Activate the **Move** tool to move the character cube. You will notice that the entire character moves with the character cube. This is because the character root joint position is defined in terms of its relationship to the character cube. If required, you can also use the **Rotate** tool to rotate the character cube. This can be required if your animation pipeline requires a change in the orientation of the local character coordinate system.

   You can also set the position and orientation of the character cube by using its **Position And Orientation** settings in the Property Editor.
4. Once you have set the correct position and orientation of the character cube, you must now restore the character to its original position and orientation. To do so, select the character **root node** in the viewport or Node View. Use the Move and Rotate tools to reposition the root node. Alternatively, you can adjust the **Position And Orientation** settings of the root node using the Property Editor.

   When editing the **root joint** position and orientation, keep in mind that you need to make the **opposite** changes to the position and orientation that you made to the **character cube**. For example, if you increased the Y position of the character cube by one metre, you will need to decrease the Y position of the character root node by one metre in order to offset this change.

Editing joint hierarchies

One of the most common editing tasks in Character Edit Mode is to change the lengths and orientations of the joints in a **simulation** character to match its corresponding **reference** character. You will not usually modify the joints in a reference character, since reference characters usually reflect an external character hierarchy.

You can edit joint lengths using the Move or Scale tools. You can edit joint orientations using the Move and Rotate tools.

The process of editing the length and orientation of joints in custom characters is often called **resizing** a character. When you change joint lengths and orientations, you are editing the character definition, which also defines its **default pose**. When you add a character to a scene, or use the **Reset Pose** command, the character is displayed in its default pose.

Keep in mind that *endorphin* simulations works optimally with human-sized characters. It is generally not advisable to create characters that are unusually large or small. As a rough guide, ensure that characters are not more than double the size, or less than half the size, of an average-sized human.

Note Do **not** use the Pose Move or Pose Rotate tools to edit joints in Character Edit Mode. You should only use the pose tools when you want to test a character.

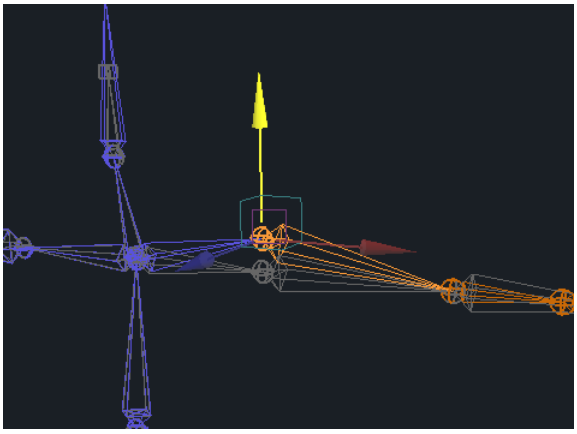
Joint lengths

The **length** of a joint is defined as the distance from its connector to its child joint connector. To change the length of a joint, you do not move the joint itself. Instead, you move its **child** joint using the Move tool. This effectively changes the length of the joint along that direction.

Keep in mind that some joints have **multiple** child joints. For example, the root joint has **three** child joints: LeftHipJoint, RightHipJoint and LowerSpineJoint. You can move any of these child joints using the Move tool to edit the length of the joint in the direction of that child joint.

When you change the length of a joint, any **child** mass objects or collision objects connected to the joint are also resized. For example, if you double the length of a joint, its child mass and collision objects will also double in length. You can separately edit the size and shape of child mass objects and collision objects. It is good practice to edit the joint length first, and then to edit mass and collision objects.

Keep in mind that when you change a joint length, you will affect the position of all the subsequent child joints down the rest of the joint hierarchy.

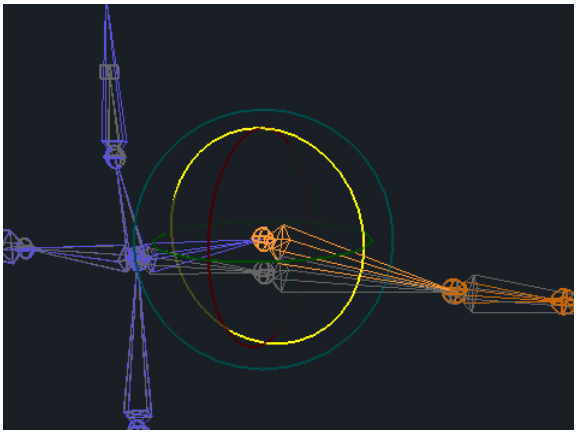


Joint orientations

The **orientation** of a joint is defined as the position of its **joint limit offset indicator** relative to its joint limit. To change the orientation of a joint, you rotate the joint using the Rotate tool. Alternatively, you can change the orientation of a joint by moving its child joint using the Move tool. However, this approach can only be used when a joint only has one child joint. When a joint has multiple child joints, such as the UpperSpineJoint, you can only use the Rotate tool to change the joint orientation.

It is important to ensure that the new orientation of the joint limit offset does not fall outside the joint limit. If the joint falls outside its joint limit, the character will not simulate well.

Keep in mind that when you change a joint orientation, you will affect the position of all the subsequent child joints down the rest of the joint hierarchy.



Creating matching joint hierarchies

The process of editing the length and orientation of joints in custom characters is often called **resizing** or **reshaping** a character.

A common task when editing custom characters is to change the length and orientations of joints in a simulation character to match a corresponding reference character as closely as possible. When you are moving joints in Character Edit Mode, you can **snap** simulation character joints to reference character joints. Use joint snapping when you want to exactly align a simulation character joint to reference character joint.

Differences between character joint hierarchies

The joint hierarchy in a custom simulation character will usually not match the joint hierarchy in a reference character exactly. The individual joint lengths will almost certainly be different. In addition, there will usually be different numbers of joints. There are many common ways in which character hierarchies might differ:

- **Spine:** Some characters only have one or two joints along the spine, whereas other characters may have a joint corresponding to every vertebral joint. *endorphin* simulation characters have three spine joints between the pelvis and the neck.
- **Forearms:** Most characters only have arm joints for the shoulder, elbow and wrist. *endorphin* simulation characters also include a joint midway along the forearm, which acts as a forearm twist joint.
- **Feet:** Most characters have relatively simple joint arrangements for the feet. *endorphin* characters have quite complex feet, and include joints not only for the ankles and toes, but also an additional mid-foot joint to model foot flex. Articulated feet are necessary in the simulation character to help ensure high-quality foot-ground interactions with minimal foot slip.

Scaling reference characters to match simulation character dimensions

When you a working with a simulation-reference character pair, you need to match the simulation character joint positions as closely as possible to the reference character joint positions. **However, you cannot significantly change the absolute size of the simulation character.** If you adjust the height of simulation character outside the range about 1.5 to 2.5 metres (4 feet to 8 feet) you may find that behaviours start to work less well.

If your reference character size is outside this range—either because your character really **is** very large or very small, or because your reference character length units are not in metres—you can **scale** the reference character so that it matches the size of the simulation character. This scale factor is used when you use Auto Motion Transfer to transfer motion between your simulation and reference characters.

Reshaping simulation characters to reference characters

When you are working with a simulation-reference character pair, you will nearly always need to **reshape** the simulation character joint hierarchy to match the reference character joint hierarchy. The aim is to have as many joints as possible in the simulation character accurately positioned at the locations of the corresponding reference character joints. Sometimes, this will not be possible:

- In some cases, a **simulation** character will have a joint that does not exist in the reference character. For example, the forearm twist joints in the simulation character (LeftForearmTwistJoint and LeftForearmTwistJoint) will often not have a corresponding joint in the reference character. In these cases, you will typically position these joint so that they are midway and linear between the corresponding elbow and wrist joints.
- In other cases, a **reference** character will have a joint that does not exist in the simulation character. For example, some reference characters often have more spine joints than the simulation character. In these cases, you might want to edit the simulation character joints to align them with their closest matching reference character joints, and leave other joints unmatched. Alternatively, you might edit a set of simulation character joints, such as all the spine joints, and distribute them equally without matching any of the reference character joints.

  For example, the simulation character has three spine joints. If the reference character has four spine joints, you could either **(a)** associate the simulation joints with three of the reference joints, and skip one of them; or **(b)** equally space the three simulation joints along the spine of the reference character so that the first and last joint positions match, and the intermediate simulation character joint lies part-way between the middle two reference character joints.

Resizing simulation characters to reference characters

When you a working with a simulation-reference character pair, you will nearly always need to **resize** the simulation character so that its mass objects and collision objects reflect the size and mass of the reference character. It is usually good practice to carry out mass and collision object reshaping **after** you have first aligned the character joint hierarchies.

Scaling reference characters

When you are working with a simulation-reference character pair, you need to match the simulation character joint positions as closely as possible to the reference character joint positions. **However, you cannot significantly change the absolute size of the simulation character.** If you adjust the height of the simulation character outside the range about 1.5 to 2.5 metres (4 feet to 8 feet) you may find that behaviours start to work less well.

If your reference character size is outside this range—either because your character really **is** very large or very small, or because your reference character length units are not in metres—you can **scale** the reference character so that it roughly matches the size of the simulation character. After you have roughly scaled the reference character to match the simulation character, you can then begin the detailed process of precisely matching the simulation character to match the reference character.

The reference character stores its scale factor in its **.nmc** character file. This scale factor is used when you use **dynamic motion transfer** to transfer motion between your simulation and reference characters.

To scale a reference character in Character Edit Mode:

1. Open the simulation-reference character pair in Character Edit Mode.
2. Activate the reference character by clicking the **Activate Reference Character** button on the Main Toolbar. Alternatively, press **Ctrl+Tab** to toggle the active character.
3. The reference character has a **Scale** property in the Property Editor. The default scale factor is 1.0. This scale factor is stored in the reference character **.nmc** file.

   Note Simulation characters and prop characters also have character scale factors displayed in the **Property Editor**. However, these character types always have a character scale factor of 1.0, and this value cannot be edited. The character scale factor can only be edited for reference characters.
4. You can also edit the reference character scale factor using the **Scale** tool.

   Activate the **Scale** tool and select the reference character's **character cube** in the viewport, or select the reference character in the Node View. Scaling the character will recursively scale all the joint lengths of the character, as well as its mass and collision objects. The character scale factor will be updated in the Property Editor as you scale the character.

Using the reference character scale factor

If you have changed the reference character scale factor, you must use this scale factor when you use the reference character to transfer motion onto the corresponding simulation character, and vice versa.

For example, suppose your source character rig was defined in terms of **inches**, rather than metres. When you first created a simulation-reference character pair, you would have found that your reference character was imported 39.37 times bigger than it should have been—since there are 39.37 inches in one metre. Applying a reference character scale factor of 0.0254—which is the inverse of 39.37—will correct for this scale factor.

Later, when you are editing a scene that uses the simulation character, you might import motion onto your simulation character using Auto Motion Transfer. If you locate the **Transfer From Reference Character** setting in the **Import Options** dialog, you will notice that the **Scale** setting is automatically updated to a value of 0.0254—this value has been retrieved from the reference character file, and ensures that reference character motion is correctly converted from inches to metres before it is dynamically mapped to the simulation character.

Similarly, if you use Auto Motion Transfer to export motion from your simulation character, you should find that when you turn on the **Transfer To Reference Character** setting, the **Scale** setting is automatically updated to a value of 39.37.

Moving, rotating and scaling joints

One of the most common editing tasks in Character Edit Mode is to change the lengths and orientations of the joints in a **simulation** character to match its corresponding **reference** character. You will not usually modify the joints in a reference character, since reference characters usually reflect an external character hierarchy.

You can edit joint **lengths** using the Move or Scale tools. You can edit joint **orientations** using the Move or Rotate tools.

The process of editing the length and orientation of joints in custom characters is often called **resizing** or **reshaping** a character. When you change joint lengths and orientations, you are editing the character definition, which also defines its **default pose**. When you add a character to a scene, or use the **Reset Pose** command, the character is displayed in its default pose.

Note that you usually do **not** use the Pose Move or Pose Rotate tools to edit joints in Character Edit Mode.

To move a joint:

1. Ensure the desired character is editable. In most cases, you will be editing **simulation** character joints. To edit the simulation character, click the **Activate Simulation Character** button  in the Main Toolbar.
2. Activate the **Move** tool. Select **Edit > Move**, or click the **Move** button  in the Main Toolbar. The keyboard shortcut is **Q**.
3. Select the joints you want to move. You can select multiple joints, but it is usually preferable to edit a single joint at a time. You can use the **Mirror** tools to copy any changes you make to the other side of the character. If you are editing a simulation-reference character pair, you can also use the Move tool to precisely **snap** joints in order to exactly align them.
4. Use the Move tool to move the joint. You can use all the standard functionality of the Move tool in Character Edit Mode, such as free movement and constrained movement.
    - When you move a joint, you do not affect its own length or orientation. Rather, you affect the **length** and **orientation** of its **parent** joint. This is because the length and orientation of a joint are defined by the relative positions of the joint and its child joint. Note that if the parent joint has multiple child joints, then moving a child joint does not change the orientation of the parent joint.
    - Although you can use the Move tool to modify joint orientations, but it is usually easier to use the Rotate tool to edit orientations. It is important to ensure that any new orientation of a joint does not fall outside its joint limit. If a **joint limit offset indicator** is positioned outside its joint limit, the character will not simulate well.
    - It is good practice to display relevant joint limits and joint limit offset indicators in Character Edit Mode so you can visually confirm that any orientation changes you make are valid.
    - When you move a joint, its child mass objects and collision objects are also moved.

To rotate a joint:

1. Ensure the desired character is editable. In most cases, you will be editing **simulation** character joints. To edit the simulation character, click the **Activate Simulation Character** button  in the Main Toolbar.
2. Activate the **Rotate** tool. Select **Edit > Rotate**, or click the **Rotate** button  in the Main Toolbar. The keyboard shortcut is **W**.

3. Select the joints you want to rotate. You can select multiple joints, but it is usually preferable to edit a single joint at a time. You can use the **Mirror** tools to copy any changes you make to the other side of the character.
4. Use the **Rotate** tool to rotate the joint. You can use all the standard functionality of the Rotate tool in Character Edit Mode, such as free movement and constrained movement. However, do **not** change the rotation pivot point.
5. When you rotate a joint, you are moving the joint **relative** to its joint limit. You are **not** changing the orientation of the joint limit itself. This is an important distinction to make.
    - It is important to ensure that any new orientation of a joint does not fall outside its joint limit. If a joint limit offset indicator is positioned outside its joint limit, the character will not simulate well.
    - It is good practice to display relevant joint limits in Character Edit Mode so you can visually confirm that any orientation changes you make are valid.
    - When you rotate a joint, its child mass objects and collision objects are also rotated.

To scale a joint:

1. Ensure the desired character is editable. In most cases, you will be editing **simulation** character joints. To edit the simulation character, click the **Activate Simulation Character** button  in the Main Toolbar.
2. Activate the **Scale** tool. Select **Edit > Scale**, or click the **Scale** button  in the Main Toolbar. The keyboard shortcut is **E**.
3. Select the joints you want to scale. You can select multiple joints, but it is usually preferable to edit a single joint at a time. You can use the **Mirror** tools to copy any changes you make to the other side of the character.
4. Use the Scale tool to scale the joint. You can use all the standard functionality of the Scale tool in Character Edit Mode.

    When you scale a joint, you affect its **length**, but not its orientation. Its child mass objects and collision objects are also scaled.

To scale a joint hierarchy:

- If you want to scale a joint, along with all its child joints down the hierarchy, hold down the **Ctrl** key while scaling the parent joint. This **recursive joint scaling** is also available if multiple joints are selected.

    For example, if you wanted to extend every joint in both arms of the standard simulation character, you could hold down the **Ctrl** key and select the **LeftClavicleJoint** and **RightClavicleJoint**. You could then activate the Scale tool,

and continue to hold down the Ctrl key while scaling the selected joints. The child joints down both arms will also be scaled—along with any of their child mass and collision objects.

Snapping joints

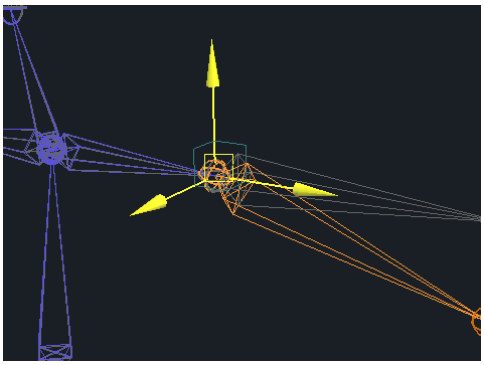
When you are editing a simulation-reference character pair, and you are using the Move tool to adjust the positions and orientations of character joints, there is an additional **snapping** mode to help you accurately align joints. You will nearly always use joint snapping to snap joints on the simulation character to joints on the reference character.

Using the snap tool

When you **snap** one joint to another, the position of the joint is adjusted to exactly match the position of the other joint that is being snapped to. Usually, you will be snapping joints of the simulation character to their corresponding joints in the reference character.

Once a joint has been snapped into position, it will **keep** that position unless you use the Move tool to move it away from that position. This is very useful, as it means you can snap joints in **any sequence** you like. For example, you can snap the shoulder joints together, and then snap the wrist joints together, and then move or rotate the elbow joint without interfering with the positions of the shoulder or elbow joints. This feature was first added in *endorphin* 2.6. In previous versions of endorphin, snapping joints would **not** create a permanent connection.

You can use the snap tool to snap any joint, including the skeletal root joint. In versions prior to *endorphin* 2.6, it was not possible to snap skeletal root joints.



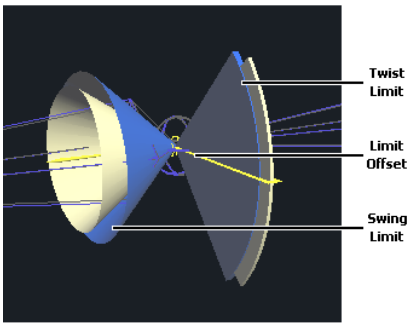
To snap character joints:

1. Ensure the simulation and reference characters are both visible. If the simulation character is not visible, right-click and select **Show/Hide Character > Simulation Character**. If the reference character is not visible, right-click **Show/Hide Character > Reference Character**.
2. Activate the reference character by clicking the **Activate Reference Character** button  in the Main Toolbar.
3. Select a joint in the simulation character to display the Move manipulator.
4. Move the joint using **free** movement by clicking inside the smallest **purple box manipulator** and dragging the mouse. Do not move the joint using a linear or planar constrained drag.
5. Move the joint towards a corresponding joint in the reference character that you would like to snap to.
6. When your **cursor** is moved to within the **snapping threshold** of the corresponding reference character joint, the simulation character joint that you are dragging will be **snapped** to the reference character joint. You will see the simulation character joint move so that it exactly superimposes the reference character joint.

   You can also snap reference character joints to simulation character joints, although typically you will be editing simulation characters to match reference characters, rather than vice versa.
7. If you continue to drag, you can **unsnap** the simulation and reference character joints by moving the mouse beyond the snapping threshold.

Working with joint limits

Most joints have a corresponding **joint limit**. The joint limit specifies the range of movement available. **Skeletal joints** have separate limits for the twist and each of the swing directions. **Hinge joints** have a limit for the hinge axis. **Simple joints** are not articulated and so do not have a joint limit. Likewise, the **root joint** does not have a joint limit.



In Character Edit Mode, you can perform two distinct types of editing that affect joints and joint limits:

- You can edit the **orientation of joints** with respect to their joint limits by using the Move and Rotate tools to edit joints. This changes the default orientation of joints but does not change the joint limits. For example, if you wanted to change the default pose of your custom character so that its arms point forwards rather than outwards, you would typically edit the joint orientations.
- You can edit the **orientation of joint limits** themselves using the Rotate tool. When you change the orientation of a joint limit, you are changing the allowable range of motion for that joint. For example, if you were creating a primate-like character such as a gorilla, you might reorient the shoulder joint limits downwards so that the average range of motion points reflects that of a more primate-like body shape.

  You can also edit other **properties of joint limits** using the Property Editor. For example, you can modify the orientation, the hard and soft limit ranges, and stiffness and damping values for joint limits using the Property Editor.

  One common reason to modify joint limits, particularly their orientations and hard limit ranges, is to fine-tune the motion generated when you apply adaptive behaviour events to the character. Behaviours can be quite sensitive to joint limit orientations and ranges, and modifying character joint limits can be another way of varying the resulting animation.

Joint limit offsets

Joint limits do not actually directly constrain the range of motion of their corresponding child joints. Rather, they constrain the range of their corresponding **joint limit offset**. The child joint itself has a relative offset to the joint limit offset. Hinge joints have a single hinge joint limit offset. Swing joints have both swing and twist limit offsets.

Swing joint limit offsets

Skeletal joints have **swing joint limit offsets**, which are associated with their swing joint limits. By default, swing joint limit offsets are aligned axially with their child joints. If you change the orientation of a joint using the Move or Rotate tools, the corresponding parent swing joint limit offset is also reoriented to maintain the existing relationship, which is typically axial. In principle, you are able to use the Rotate tool to rotate a swing joint limit offset so that it is no longer axial with its corresponding child joint. However, we do **not** recommend this, as it can make the character joint definitions confusing.

Twist joint limit offsets

Skeletal and hinge joints both have **twist joint limit offsets**. Unlike swing joint limits, it is useful to **rotate** twist joint limits. You will need to rotate a twist joint limit offset if you want to keep the same twist range, but change its relative position relative to the child joint. For example, suppose a joint has a twist range of 90 degrees, made up of twist ranges of –45 degrees and +45 degrees. If you want to keep the twist range at 90 degrees, but change the relative ranges to –20 degrees and +70 degrees, you would need to rotate the twist joint limit offset by 25 degrees.

If you want to use the Rotate tool to rotate the twist joint limit offset, you should always rotate the offset about its joint axis. To do so, first toggle the rotate coordinate system to use the local coordinate system by selecting **Edit > Toggle Coordinate System**. Next, ensure that you use the **red** circular manipulator to rotate about the **local X axis**.

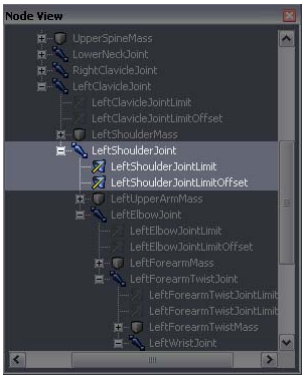
Alternatively—and possibly more easily—you can edit the Twist Orientation property of the corresponding joint limit offset directly using the Property Editor.

Displaying joint limits

Like other objects in the virtual world, you can toggle the **visibility** of joint limits and joint limit offsets in the viewports. However, unlike most other object types, joint limits and joint limit offsets are generally **not** shown by default. The reason for this is partly to reduce visual clutter in the viewports, and partly to make it more difficult to inadvertently edit joint limits.

To display a joint limit or a joint limit offset:

1. **Select** the corresponding joint in the viewport or in the Node View.
2. Locate the two child items immediately below the corresponding joint in the Node View. They will have **joint limit icons**. One of the items will have **– JointLimit** suffix. This is the joint limit itself. The other item will have a **– JointLimitOffset** suffix. This is the joint limit offset.
3. **Click** the icons of each of these items to display the corresponding item in the viewports. The Node View are icons are solid when these items are visible in the viewports, and are partially transparent when these items are hidden in the viewports.
4. If you do see the joint limit or the joint limit indicator appear, select **View** and ensure that **Joint Limits** and **Joint Limit Offsets** are both turned on. Alternatively, **right-click** in the viewport and ensure that **Joint Limits** and **Joint Limit Offsets** are selected. This setting is part of the viewport display filter that determines what types of entities are displayed in the viewport.



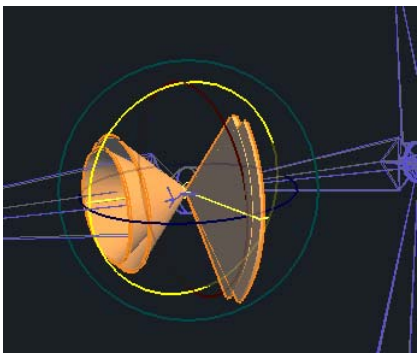
Rotating joint limits

You can edit the **orientation** of joint limits and joint limit indicators using the Rotate tool. When you change the orientation of a joint limit, you are changing the allowable range of motion for that joint with respect to the corresponding joint limit.

To rotate a joint limit:

1. Activate the **Rotate** tool. Select **Edit > Rotate**, or click the **Rotate** button  in the Main Toolbar. The keyboard shortcut is **W**.
2. **Select** the joint limit you want to rotate. You can use the viewport or the Node View to select the joint limit. Most of the time, you will be rotating the **swing joint limit** cones of skeletal joints.
3. Use the Rotate tool to rotate the joint limit. You can use free rotation, or constrained planar rotation. You should **not** edit the rotation pivot point.
4. When you are editing a swing joint limit, ensure that your changes do not result in the joint limit offset indicator falling outside allowable joint limit range. If it does so, you will either need to rotate the joint or the joint limit to ensure that the indicator is in the allowable range.

   Alternatively, you may find it easier to select the corresponding joint limit using the viewport or Node View, and then modifying its **Swing 1 Orientation**, **Swing 2 Orientation** and **Twist** properties in the Property Editor.



To rotate a twist joint limit offset:

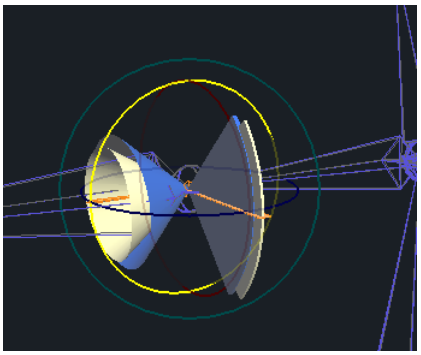
1. Activate the **Rotate** tool. Select **Edit > Rotate**, or click the **Rotate** button  n the Main Toolbar. The keyboard shortcut is **W**.
2. **Select** the joint limit offset you want to rotate. You can use the viewport or the Node View to select the joint limit. Keep in mind that you should only rotate twist joint limit offsets. You should **not** rotate swing joint limit offsets—this will result

in the swing limit offset indicator no longer aligning axially with the joint child, and will make editing the character confusing.

3. Ensure that the Rotate tool is rotating about local coordinates rather than global coordinates. Select **Edit > Toggle Coordinate System** to toggle between local and global coordinates. You will see the orientation of the Rotate tool manipulators change as you toggle between these two modes.

4. Use the **red circular manipulator** of the Rotate tool to rotate the joint limit offset about the **local X axis** of the corresponding joint. This ensures that the twist limit offset is modified, without modifying the swing limit offset.

   Alternatively, you may find it easier to select the corresponding joint limit offset using the viewport or Node View, and then modifying its **Twist Orientation** property in the Property Editor.



Editing mass and collision objects

In Character Edit Mode, you can change the size, shape and properties of the **mass** and **collision** objects of a character. You cannot add or remove mass objects. However, you can add and remove collision objects.

Automatic updates after editing joints

When you edit joint lengths in Character Edit Mode, child mass and collision objects of the joint are updated **automatically**. For example, if the length of the joint is doubled, the lengths of the corresponding child mass and collision objects is also doubled. Note that this applies to cylinder, sphyl and box objects, but not to sphere objects.

Automatic updating of mass and collision objects is very useful. With automatic updating, you can confidently reshape the character joint hierarchy, and have the new character shape reflected in the positions and sizes of corresponding mass and collision objects.

However, automatic updating does establish a workflow in which you should edit the joint hierarchy **first**, before editing the mass or collision objects. This is to avoid

inadvertently making changes to mass or collision objects by subsequent changes to the character hierarchy.

Editing mass and collision objects

In Character Edit Mode, you can edit the positions, orientations and sizes of mass objects and collision objects using the Move, Rotate and Scale tools. These tools work in the same manner as the corresponding tools when you are editing a scene. When you are editing the position and orientation of mass and collision objects, keep in mind that you are editing their positions with respect to their corresponding parent joints.

You can also set the **properties** of individual mass and collision objects using the Property Editor, in the same manner as when editing a scene. The only property that is still read-only in Character Edit Mode is the name of each object. The **undo** and **redo** commands also work in Character Edit Mode, with the same limitations as apply when editing scenes.

Some common changes to mass and collision objects include:

- Editing the **mass** of mass objects. This is useful for creating lighter or heavier characters, and for changing the dynamics of a character. You might change the masses to reflect a distribution of muscle, fat or armour. You can change the overall mass of a mass object by changing its size or density.
- Editing the **material** of collision objects. This is particularly useful for changing the dynamics of interactions. You might change the surface friction to reflect the smooth, slippery surface of bare skin or the rough surface of chain armour.

Adding and removing collision objects

You can add and remove collision objects from a joint. You cannot add or remove mass objects.

Adding new collision objects is a very important part of the character editing process. You can build up a musculature for stronger characters or add feminine curves for female characters. You can also add helmets, armour, backpacks and other accessories.

The **Car Hit.ens**, **Gymnast Rings.ens** and **Tackle.ens** sample scenes all include samples of custom simulation characters that have been reshaped with additional collision objects to reflect additional muscle bulk. In addition, the characters in **Tackle.ens** also have collision objects that define helmets and shoulder pads. You can find the sample scenes in **Resources\Scenes\Sample Scenes**.

Setting collision object materials

You can change the way a character interacts with itself, and with other characters and objects, by modifying its collision object materials. By default, characters use the following material types. You can change the material of any collision object. Keep in mind that in a scene, you can also change the properties of the **interaction** between two materials.

- Most of the collision objects belonging to the standard simulation character have the **skin** material type. This ensures that body parts can move past each other relatively smoothly.
- The fingers of the standard simulation character have the **sponge** material type, which ensures that the fingers are both soft yet grippy.
- The heel, ball and toes of the character's feet have the **rubber** material. This ensures that the feet have a grippy, bouncy interaction with the ground.

Adding and removing collision objects

In Character Edit Mode, you can change the size, shape and properties of the mass and collision objects of a character. You can also add and remove collision objects. Note that you cannot add or remove mass objects.

To add collision objects to a character:

1. Specify one or more **mass objects** that will act as the parents of the new collision object. All collision objects have a corresponding parent mass object. You can use the viewports or the Node View to select the parent mass object or objects.

   You can directly specify the parent mass object by selecting it. You can also indirectly specify the parent mass object by selecting a **collision object** or **graphical object**—in these cases, their **parent** mass objects are used as the parents of the new collision object.

   Similarly, you can also indirectly specify the parent mass object by selecting a **joint**—in this case, its **child** mass object is used as the parent of the new collision object.

2. Select **Environment > Create Sphyl Collision Object** to add a new sphyl-shaped collision object to the selected mass object. You can also click the **Create Sphyl Collision Object** button in the Main Toolbar.

   There are similar commands and buttons to create cylinder, box and sphere collision objects.

   If you had selected multiple mass objects, then multiple collision objects will be created, with each new collision object owned by a different selected mass object.

To remove collision objects from a character:

1. **Select** one or collision objects. You can use the viewport or the Node View to select the collision objects.
2. Select **Edit > Delete** to delete the collision objects. The keyboard shortcut is **Delete**. There are no limitations on deleting collision objects. You can delete **all** the collision objects associated with a given mass object.

Setting collision rules for collision objects

By default, collision objects collide with all other collision objects, including other collision objects belonging to the same character. However, characters are usually composed of a large number of collision objects, many of which may intersect.

Typically, we will want to prevent some pairs of collision objects—typically those that are intersecting in the default character pose—from colliding with each other. At the same time, we need to ensure that other pairs of collision objects—such as arm and torso collision objects—continue to collide with each other, to avoid parts of a character from self-penetrating. The key is to correctly set the **collision rules** for each character.

It is important that any pairs of intersecting collision objects—other than collision objects belonging to the **same** mass object—have their collision rules set so that they do not collide. Otherwise, the character will rapidly spring apart when simulated in a scene. You can test for the existence of incorrectly-assigned collision rules by using the **Pose Move** or **Pose Rotate** tools in Character Edit Mode. These tools simulate the character, and allow you to test the validity of your collision rules.

Collision objects of the same mass object

Collision objects belonging to the same mass object **never** collide with each other. For example, the **HeadMass** object of the standard simulation character has three collision objects that interpenetrate—the **HeadCollision01**, **HeadCollision02** and **HeadCollision03** objects. These collision objects will never collide with each other.

Collides With Parent setting

To prevent the collision objects of a mass object from colliding with the collision objects of the mass object of its **parent joint**, turn off the **Collides With Parent** setting for all the child collision objects.

Most of the collision objects in the standard simulation character have the Collides With Parent setting turned off. For example, the **LeftUpperArmCollision01** collision object has this setting turned off so that it does not collide with the **LeftShoulderCollision01** collision object. Note that a few collision objects—such as the **LeftThumbCollision01**

object—have this setting turned on. In the case of the thumb, it allows the thumb to collide with the **LeftForearmTwistCollision01** object.

Collides With Siblings setting

To prevent the collision objects of a mass object from colliding with the collision objects of its **sibling mass objects**—that is, other mass objects whose parent joints share a common parent—turn off the Collides With Sibling setting for all the child collision objects.

For of the collision objects in the standard simulation character have the Collides With Siblings parent turned off. This is particular true for all the collision objects making up the **torso** of the character. However, it is important that the character's **legs** can collide with each other—and as you will note, the **LeftUpperLegCollision01**, **RightUpperLegCollision01**, **LeftLowerLegCollision01** and **RightLowerLegCollision01** objects do have this setting turned on.

Collides With Ancestors setting

To prevent the collision objects of a mass object from colliding with the collision objects of **any** of the mass objects that are higher up in the character hierarchy—turn off the Collides With Ancestors setting for all the child collision objects.

Many of the collision objects of the standard simulation character have this setting turned on. However, collision objects belonging to the forearm, hand and head of the character generally have this setting turned on. This ensures that the forearms, hands and head can collide with the torso of the character.

Using helper graphical objects

You can import **Wavefront OBJ mesh files** into a scene, or onto a character in Character Edit Mode, to create partially transparent helper graphical objects.

Helper graphical objects are a useful aid when you are editing a scene or character. For example, in Character Edit Mode it is useful create helper graphical objects by importing the character skin as an OBJ mesh file. You can then use this graphical object as a template when creating and editing corresponding collision objects. This is often called **filling out** the character.

Graphical objects are displayed using the grey **graphical objects** colour. By default, this colour is only 60% solid, which means that you can also see other objects that may be inside the graphical object.

Attaching graphical objects to reference characters

If you are using a helper graphical object to help match a simulation character to a reference character, you should select the reference character **before** importing the OBJ mesh file. This will create the new graphical object as a child of the reference character. You can add multiple graphical objects to a character.

When graphical objects are attached to the reference character, you can work with them in two ways:

- To use graphical objects as helper objects, activate the **simulation** character. This freezes the reference character, including any graphical objects that are attached to it. The graphical objects are still visible, but can no longer be selected. This allows you to edit the simulation character, using the graphical objects as a reference when positioning mass objects and collision objects.
- To move, scale or rotate graphical objects, activate the **reference** character. Then use the Move, Rotate and Scale tools to adjust the graphical object.

To create a helper graphical object:

1. Select the desired parent object of the graphical object. If no objects are selected, the new graphical object will be created as a child of the environment character. If a joint, mass object or collision object is selected, the new graphical object will be created as a child of that selected object.

   If you are using a helper graphical object to help match a simulation character to a reference character, you should select the **reference** character.

2. Select **File > Import…**. The keyboard shortcut is **I**.

3. The **Select File To Import** dialog appears. Set the file type filter to **OBJ Files (*.obj)**, and browse for a desired OBJ mesh file. When you have found the desired OBJ mesh file, click **Open**.

4. The **Import Options** dialog appears. You can set the units of the file to be either metres, centimeters, millimeters, feet or inches. You can also set an optional scale factor, and also specify whether the file considers the Y-Axis or Z-Axis to be the vertical axis.

5. Click **OK** to import the OBJ mesh file as a new graphical object.

6. You can use the Move, Rotate and Scale tools to change the position, orientation and size of the new graphical object.

Creating symmetric characters

Most of the custom simulation characters that will create will naturally be fairly **symmetric**. When a character is symmetric, the left-hand side of the character will be identical to its right-hand side.

Joint lengths, orientations and limits are nearly always symmetric. Mass objects and collision objects are usually also mostly symmetric, although sometimes you might want to add asymmetries to reflect additional objects that might affect the shape or size of a character, such as weapons, armour or baggage.

Mirror tools

In Character Edit mode, the **Mirror tools** can help make it easier to model symmetric characters by letting you quickly copy one side of a character to its other side without having to manually model both sides.

The Mirror tools work on the basis of **named item pairs**. For example, the LeftShoulderMass and RightShoulderMass mass objects form such a named item pair. As long as the corresponding named item can be identified, the Mirror tools can copy across the parameters of the source item, such as its size, position and orientation, to the target

item on the other side. The actual location or pose of your character generally does not affect the Mirror tools.

Mirror plane

Once *endorphin* has identified pairs of mirror items, the actual mirroring occurs about the YZ plane, passing through the origin. This means that you should only mirror characters that are centred at the origin, and facing along the Z axis.

Mirroring new collision objects

If you have created or removed collision objects, these changes are reflected when you use the Mirror tool. For example, if you add collision objects to the left arm in order to define its musculature, the Mirror Left To Right command will copy these additional collision objects to the right arm.

Mirroring reference characters

The mirror tools are only available in Character Edit mode. You will mainly use them when editing custom simulation characters. You can also use them with reference characters, but **only** if the reference character naming convention includes the text "left" and "right". This naming requirement is not case-sensitive or position-sensitive, so pairs such as "leftArm" and "rightArm" or "LEG_LEFT" and "LEG_RIGHT" will mirror correctly. However, pairs such as "lhand" and "rhand" will not be mirrored.

Mirroring objects

In Character Edit mode, the **Mirror tools** can help make it easier to model symmetric characters by letting you quickly copy one side of a character to its other side without having to manually model both sides.

To mirror the entire character:

- To mirror objects from the left-hand side of a character to its right-hand side, select **Character > Mirror Left To Right**. The keyboard shortcut is **Alt+LeftArrow**. Alternatively, click the **Mirror Left to Right** button  in the Main Toolbar.
- To mirror objects from the right-hand side of a character to its left-hand side, select **Character > Mirror Right To Left**. The keyboard shortcut is **Alt+RightArrow**. Alternatively, click the **Mirror Right To Left** button  in the Main Toolbar.

  These commands copy the size, position and orientation of all joints, mass objects and collision objects from one of the sides of the character to the

corresponding objects on its other side. In addition, properties such as joint limits, mass object densities and collision object modes are also copied.

If the **Auto Create Connections** setting is turned on in the Motion Transfer Editor, connections between mass objects and rig nodes are also mirrored.

To mirror selected objects only:

1. Select the objects that you want to mirror. You can select joints, joint limits, mass objects or collision objects. Generally you select objects from only **one** side of the character.
2. To mirror the selected objects from the left-hand side of a character to its right-hand side, select **Character > Mirror Selected Left To Right**. The keyboard shortcut is **Ctrl+Alt+LeftArrow**. Alternatively, click the **Mirror Selected Left to Right** button  in the Main Toolbar.

   To mirror the selected objects from the right-hand side of a character to its left-hand side, select **Character > Mirror Selected Right To Left**. The keyboard shortcut is **Ctrl+Alt+RightArrow**. Alternatively, click the **Mirror Selected Right To Left** button  in the Main Toolbar.

   These commands copy the size, position and orientation of the **selected** joints, mass objects and collision objects from one of the sides of the character to the corresponding objects on its other side. In addition, properties such as joint limits, mass object densities and collision object modes are also copied.

   If the **Auto Create Connections** setting is turned on in the Motion Transfer Editor, connections between mass objects and rig nodes are also mirrored.

   Note If you have selected one or more joints, and then use the Mirror Selected Left To Right or Mirror Selected Right To Left commands, any child mass or collision objects of the selected joints are **also** mirrored, even if they have not been explicitly selected. Although this may seem inconsistent at first, the approach actually improves the editing workflow.

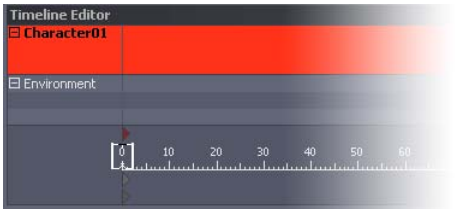
   Often, you will begin editing a character by reshaping its joints. If your viewports are in Skeletal View, you may not be aware that mass and collision objects are also being automatically updated while you editing the joints. Nevertheless, you can select desired joints and mirror them, confident that any child mass and collision object changes will be mirrored as well.

Edit Character In Scene mode

When you add a character to a scene, it is not editable by default. That is, you can move, rotate and pose the character, but you cannot change its shape or size directly. The usual way to create and edit characters is to use **Character Edit Mode**. Character Edit Mode is an environment optmised for editing characters, and includes various tools such as mirror tools that are not available when editing scenes.

However, there are times when you might want to **fine-tune** characters by editing them directly in scenes. You can do this using the **Edit Character In Scene** mode. This is a useful mode for making final adjustments to your characters in the context of the scene in which they will be used.

Keep in mind that this mode only applies to characters that you have added to a scene. It does not apply to the Environment character, since this character is always editable.



To enter Edit Character In Scene mode:

1. **Select** the character that you want to edit using the Timeline Editor, Node View or viewports.
2. Select **Character > Edit Character In Scene…**. Alternatively, right-click on the character name in the Timeline Editor and select **Edit Character In Scene…**. There is no keyboard shortcut.

   Only one character may be in Edit Character In Scene mode at any one time. When a character is in Edit Character In Scene mode, its character timeline in the Timeline Editor is displayed using the red editing character system colour. The viewport displays the label **Editing Character:** followed by the character name.

To edit a character in Edit Character In Scene mode:

Once a character is in Edit Character In Scene mode, you can perform any the following editing operations:

- Change any of the properties of any of its joints, joint limits, mass objects, collision objects or graphical objects using the Property Editor.

- Change the size and position of any of its joints, mass objects or collision objects using the **Move**, **Rotate** and **Scale** tools.
- Add new collision objects by selecting one or more of its mass objects and using one of the **Create Collision Object** commands to create new child collision objects for each of the selected mass objects.
- Remove collision objects by selected the desired collision objects and selecting **Edit > Delete** or press **Delete**.

It is important to be careful when editing characters using Edit Character In Scene mode. Some changes—such as deleting parts of the character—cannot be undone, and might prevent your character from simulating correctly.

We recommend that you only use Edit Character In Scene mode to make relatively minor adjustments to characters. Before making any changes to characters, it is good practice to make sure you have a backup of your character.

To exit Edit Character In Scene mode:

- Right-click on the character name in the Timeline Editor and select **Exit Character Editing**. Alternatively, press **Esc**.
- Various commands such as Simulate, Pose Move and Pose Rotate will automatically exit Edit Character In Scene mode.

To save edited characters to character definition files:

When you make changes to characters in a scene using Edit Character In Scene mode, the original **.nmc** character definition files are not modified. Only character instances in the **.ens** scene file are modified. To save changes you have made to characters in a scene:

1. Select the character you want to save.
2. Select **File > Export…**. The keyboard shortcut is **Ctrl+E**. This displays the **Save As** dialog.
3. Choose the **endorphin Characters (*.nmc)** file filter.
4. Enter a new character file name, or select an existing character definition file if you want to replace an existing character.
5. Click **OK** to display the **Export Options** dialog. You will not usually need to adjust any of the default export options to save the selected character.
6. Click **OK** to save the selected character to a new or existing **.nmc** character definition file.

Upgrading characters

The standard simulation character is often modified and enhanced between releases of *endorphin*. Sometimes, changes are required to the internal definition of the character in order to support newer behaviours. In other cases, improvements are designed to improve the usability of the character—such as the recent addition of elbow and knee graphical objects.

Since the standard simulation character was released with ***endorphin* 1.6**, there have been a number of standard simulation character upgrades, including the ***endorphin* 2.0**, ***endorphin* 2.5** and ***endorphin* 2.6** standard simulation characters.

If you have an existing scene that uses an older version of the simulation character—whether it is the standard simulation character, or your own custom character derived from the standard simulation character—you can **upgrade** the character in the scene so that it is based on the latest version of the character.

When you upgrade a character, *endorphin* replaces its **internal** definition with the latest simulation character definition. No part of the **external** character definition—such as its joints, joint limits, mass objects, collision objects or graphical objects—is affected when you upgrade a character in a scene.

You do **not** need to upgrade characters in existing scenes simply because a newer version of the standard simulation character has become available. However, you may find that there are additional behaviours shipped with the new *endorphin* release that cannot be used with existing characters based on older character definitions.

You can upgrade instances of characters in scenes, as well as character definitions in **Character Edit Mode**.

Note If you are upgrading a character in a scene, behaviours on the character timeline may be affected by the upgrade process. You may need to manually reset the behaviour **Type** for each of the behaviours on the character timeline.

To upgrade a character:

1. Select the character that you want to upgrade. You can upgrade any character except the Environment character. Alternatively, if you are in **Character Edit Mode**, activate the simulation character. You cannot upgrade reference or prop characters,
2. Select **Character > Upgrade Character…**.
3. The **Upgrade Character** dialog appears. This dialog displays the **internal character definition** of the selected character in the **Current Type** textbox. This dialog also displays the latest character definition available in the **Latest Type** textbox.
4. If there is a new character definition available, the **Upgrade…** button will be enabled. If the selected character internal definition already uses the latest character definition, the **Upgrade…** button will be disabled.

The internal character definition name will be one of the following values, based on the *endorphin* version in which the character was released:

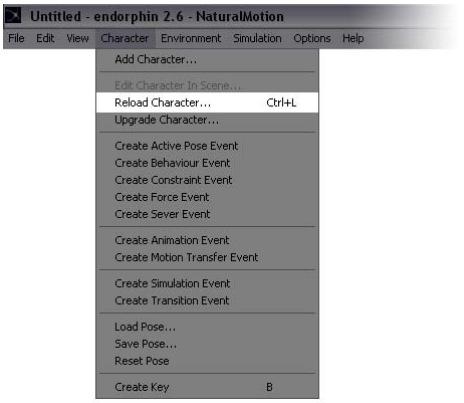
- ***endorphin* 2.6:** skeletalCharacter2.6.
- ***endorphin* 2.5:** skeletalCharacter2.5.
- ***endorphin* 2.0:** skeletalCharacter2.0.
- ***endorphin* 1.6:** skeletalCharacter.
- ***endorphin* 1.5:** stockCharacter1.5.
- ***endorphin* 1.2:** stockCharacter1.2.

5. Click the **Upgrade…** button to upgrade the selected character to use the latest internal character definition.
6. Click the **OK** or **Cancel** buttons to close the **Upgrade Character** dialog.

Reloading characters

When you add a character to a scene, endorphin creates a **copy** of the character that is stored in the **character definition file (.nmc)**. Any changes to you make to the original character definition—using Character Edit Mode—do **not** affect instances of that character in scenes. Similarly, any changes you make to characters in scenes—using Edit Character In Place mode—do not affect the character definition, or instances of that character in other scenes.

To update a character in a scene using a new character definition, you can use the **Reload Character** command. This command reads the latest version of the character stored in its character definition file, and updates the character in the scene accordingly.



To reload a character in a scene:

7. Select the character that you want to reload. You can reload any character except the Environment character.
8. Select **Character > Reload Character…**. The keyboard shortcut is **Ctrl+L**.
9. The **Reload Character** dialog appears. The **Source file** contains the full filename used when the character was added to the scene. If the character has been reloaded previously, the path contains the last-used reload file.
10. If the character definition file has been moved, or if you want to use a different character definition file altogether, click the **[…]** button to select an alternative **.nmc** character definition file.
11. Click **OK**. The selected character will be reloaded using the specified character.

    Note If you have selected a simulation character, and you reload it using a non-simulation character, such as a reference character or prop character, *endorphin* will not be able to simulate if the character timeline contains **behaviour** events. You will need to delete these events before you are able to simulate again.

Exporting characters

When you add a character to a scene, endorphin creates a **copy** of the character that is stored in the **character definition file (.nmc)**. Any changes that you make to the original character definition—using Character Edit Mode—do **not** affect instances of that chadacter in scenes. Similarly, any changes you make to characters in scenes—using Edit Character In Place mode—do not affect the character definition, or instances of that character in other scenes.

To create or update a character definition using a character in a scene, you can use the **Export** command with the **endorphin Characters (*.nmc)** file type. You can use this command to create new characters or to update existing characters.

Most of the time, you will use this command when you have modified a character in scene using Edit Character In Scene mode, and you want to update the original definition file.

You can use this command to create new characters as an alternative to Character Edit Mode, although this is generally not recommended. Character Edit Mode has been designed to make editing characters easier, and has tools such as the Mirror tools and the Move tool snap functionality that are not available when editing scenes.

To export a character definition file:

1. Select the character that you want to export. You can export any character, including the Environment character.
2. Select **File > Export…**. The keyboard shortcut is **Ctrl+E**.

3. The **Export As** dialog appears. Specify a new character definition name, or browse to select an existing character definition. Ensure that the file type filter is set to **endorphin Characters (*.nmc)**. Click **Save**.
4. Do not change any of the options in the Export Options dialog.
5. Click **OK**.

Chapter 20

Animation Pipeline

What is the *endorphin* animation pipeline?

The simplest way of working with *endorphin* is to populate a scene with simulation characters and run a dynamic simulation. However, a more powerful way of working with *endorphin* is to use it as part of an **animation pipeline**.

Once you have prepared an animation pipeline, you can:

- **Import** animation into *endorphin* scenes that you have created externally. For example, you may have existing animation data that you have generated using motion capture or keyframing. You can blend this animation into an *endorphin* dynamic simulation to create more compelling scenes.
- **Export** animation that you have generated in *endorphin* for use with other animation systems such as **Maya**, **3dsMax** and **SOFTIMAGE|XSI**. You can use many standard animation file formats such as **FBX**, **XSI** and **BVH**, and motion capture formats such as **Vicon**.

The central element of an animation pipeline is the **simulation-reference character pair.**

- The **simulation** character is a modified form of the standard *endorphin* simulation character that has been reshaped and resized to match the **reference** character.
- The **reference** character is the external character joint hierarchy that uses the animation data that you are importing or exporting.

Animation terminology

Unless stated otherwise, references to **keyframe data** generated outside of *endorphin* could easily also refer to **motion capture data**. From the perspective of *endorphin* animation import, there are no distinctions made between the various sources of animation, and they are generally grouped together as animation data in this guide.

Working with animated data

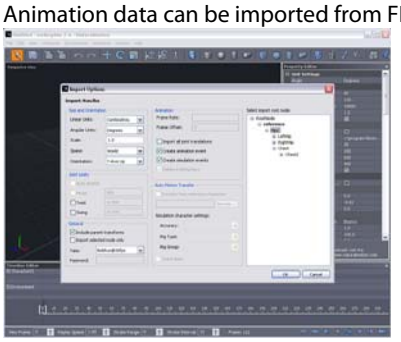
You can significantly expand the range of simulations that can be generated within *endorphin* by integrating animation data created elsewhere into your *endorphin* scenes. To use animation in *endorphin* scenes, a combination of event types are typically used together. These are: animation events, simulation and transition events, and motion transfer events.

- **Animation events** allow you to associate animation data with a character, and drive the motion of the character using this data to varying degrees.
- **Simulation events** allow you to change the simulation **mode** of a character. There are 4 modes supported by *endorphin* 2.6. Characters must be in the Full Simulation mode to be influenced by active pose, behaviour, constraint, force, sever and motion transfer events. Animation events can influence a character in all of the supported simulation modes.
- **Transition events** are similar to simulation events, except they change the simulation mode over a range of frames rather than a single frame. Typically a multi-frame transition event is used to change the simulation mode from the Full Simulation mode into one of the other modes and a single-frame simulation event is used to change the simulation mode from one of the other modes into the Full Simulation mode.
- **Motion transfer events** allow to you dynamically transfer motion from an animated source character to a simulated target character.

Introducing animation import

You can import animation data onto *endorphin* characters using a variety of standard animation formats. The standard workflow is to import animation as an **animation event** for selected character.

Animation data can be imported from FBX, XSI, BVH, AMC or Vicon V files.



Using animation data

A common use of animation is to define the initial motion of a character. For example, you might import a walk cycle or a run cycle onto a character, and then use a **simulation event** to change the simulation mode so that the character is dynamically simulated rather than fully driven by the source animation.

You can import multiple segments of animation as different animation events for the same character, and use dynamic simulation to connect them together. You will typically use simulation events when you want to blend into the Full Simulation mode from one of the other modes, and then use **transition events** for blends from the Full Simulation mode into Collision Only mode say. Simulation events are useful for immediate changes in the simulation mode, whereas transition events are useful for gradual changes in the simulation mode.

Importing onto existing characters

A common workflow is to add a character to a scene, and then to select the character and import animation data onto the character.

- If you add a custom **prop** character to a scene, such as a creature or vehicle, you can import animation directly onto that character, provided that you have animation data that is defined for the joint hierarchy of the prop character.
- If you add a custom **simulation** character to a scene, you will usually import animation onto the corresponding **reference** character, and then use **dynamic motion transfer** to transfer this motion on to the simulation character. You do not need to add the reference character to the scene explicitly. Rather, you can import motion via the reference character by using the **Auto Motion Transfer** setting in the Import Options dialog.

Importing to create new characters

You can also import animation **directly** into a scene, without specifying a target character. This creates a new prop character directly in the scene. Prior to *endorphin 2.0*, this was the only way to create new prop characters.

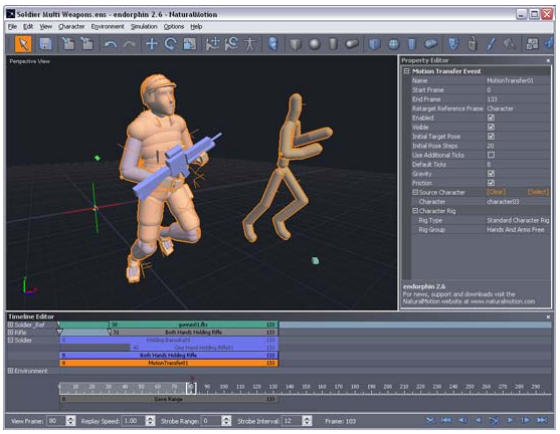
However, we recommend that you always create prop characters in Character Edit Mode, and use animation import only for the purpose of adding animation data to these characters in the scene. When you create prop characters in Character Edit Mode, you can modify their joint limits, remove unused joints, resize mass objects and collision objects, and so on. This gives you a lot more control over the behaviour of the prop character.

If you really want to create a new scene character using animation import, you can also import Acclaim ASF files and Vicon VST and VSK files. These files contain skeletal joint hierarchies, rather than animation data.

What is *endorphin* dynamic motion transfer?

Dynamic motion transfer is the process of mapping motion from a **source** character onto a **target** character during a simulation or when importing animation data into or exporting it from a scene. You will frequently use motion transfer as part of an **animation pipeline**.

To perform dynamic motion transfer, some of the mass objects of a source character are used to drive corresponding mass objects on a target character.

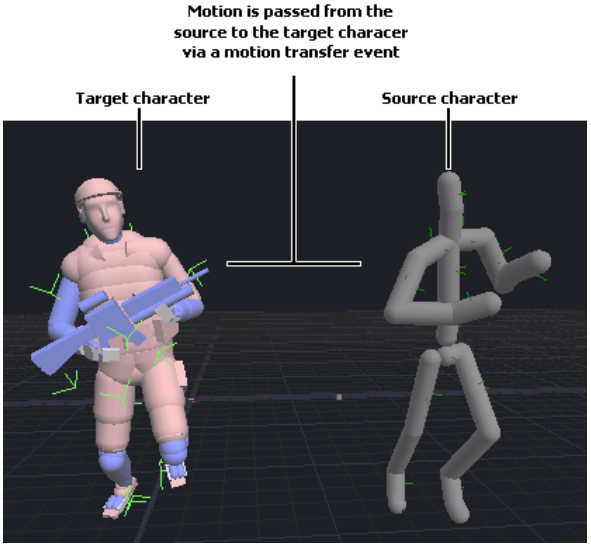


Dynamic motion transfer can be used to import existing source animation onto *endorphin* characters, and to export *endorphin* animation back out to existing characters. In addition, there are many effects that can be produced **during** an *endorphin* simulation by blending animation data with dynamic simulations using **motion transfer events**.

- When you are **importing** animation, the source character is the reference character and the target character is the simulation character. Usually, the source character will not be simulated. Instead, it will driven purely by the animation data. The target character will be simulated, and will be driven by the animated source character using dynamic motion transfer.
- When you are **exporting** animation, the source character is the simulation character and the target character is the reference character. The source character motion will not be simulated. Instead, it will be driven purely by animation data generated by an earlier simulation and stored in the frame buffer. The target character will be simulated, and will be driven by the animated source character using dynamic motion transfer.
- When you are **simulating** a scene, you can populate it with a simulation-reference character pair, and drive the reference character using an animation event. You can then use a **motion transfer event** to dynamically map this motion onto the simulation character. This technique is useful if you want to blend animation with other sources of motion, such as behaviours, active poses, forces and constraints.

Introducing character rigs

Dynamic motion transfer is the process of using the motion of a source character to drive the motion of a target character. In order to transfer motion from one character to another, you need to connect each character to a **character rig**. The motion transfer settings for each character are stored in their corresponding character rigs.



Rig nodes

A character rig is made up of a collection of **rig nodes**. Each rig node is used to connect one mass object in a source character with one mass object in a target character. You do not need to connect every mass object in the source character with every object in the target character. In fact, motion transfer can often work better when there are fewer connected mass objects.

For example, suppose you have a source character with four spine mass objects, and a target character with five mass objects. You can often connect these characters with only two rig nodes—one for the upper spine and one for the lower spine. The motion of the other spine joints can usually be realistically generated by the dynamic simulation based on the properties for the intermediate joints.

Different character rigs typically contain different numbers of rig nodes. In *endorphin 2.6*, the two main character rigs are the **Standard Character Rig** and the **Advanced Character Rig**. You should use the Standard Character Rig for dynamic motion transfer. The Advanced Character Rig will be described and enhanced in future versions of *endorphin*.

Rig node strengths

Rig nodes have corresponding position and angular **strength** values that specify the degree to which the motion of the source mass object drives the motion of the target mass object. When a rig node connection is very strong, the rig node tightly couples the motion of the source and target mass objects. When a rig node connection is not very strong, the mass objects are loosely coupled. This means that the motion of the target mass object will be more readily affected by collisions with other objects in the virtual world.

For example, you may have a walk cycle animation that you would like to use to drive a target simulation character. However, you may also want the right arm of the target character to override this motion. You can achieve this by **reducing** the positional or angular strengths of the right arm rig nodes.

You can create different sets of rig node strengths. Each set is called a rig node **group**. For example, you may have a rig node group in which all rig nodes are very strong, and another rig group in which the rig node strengths for the right arm have been reduced.

To edit character rig settings:

- Select **View > Motion Transfer Editor** to display the Motion Transfer Editor. This editor lets you view and modify the rig node connections and rig node groups for a character. The keyboard shortcut is **M**.
- You can edit rig node settings for a character **definition** in Character Edit Mode. These settings are stored in the character **.nmc** file, and are the default settings used whenever you add an instance of that character to a scene.
- You can also edit rig node settings for instances of characters in a scene. This modifies the local instance of the character in the scene, but does **not** modify the original character definition.
- If you make changes to the motion transfer settings of a character in a scene, you can **copy** these settings into its corresponding character definition file by clicking the **Save Group To Character File** button on the Strengths tab of the Motion Transfer Editor.

Unused rig nodes

By default, every rig node of the Standard Character Rig is connected to a corresponding standard simulation character mass object. Keep in mind that the reverse is not the case—there are mass objects of the standard simulation character, such as the neck, forearm and foot mass objects—that are **not** connected to rig nodes.

For a rig node to transmit motion from one character to another, it must be connected to a mass object in **both** characters. If a rig node is not connected to one of the characters, it plays no role in dynamic motion transfer, and so can be safely disconnected from the other character as well.

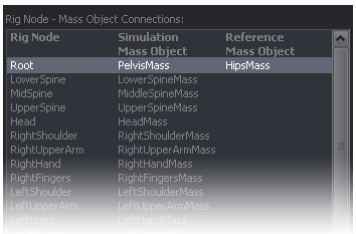
For example, you will notice that in some of the shipped custom simulation characters—such as the **AudioMotion** and **EyesJapan** characters—rig nodes such as **LeftFingers** and **LeftToes** have been disconnected from the simulation character since they are not connected to the reference character.

Connecting mass objects to rig nodes

The process of **connecting** character mass objects to character rig nodes is often called **rigging** a character.

When you create a new simulation-reference character pair, you need to rig the **reference** character, but not the simulation character. This is because new simulation characters are derived from the standard simulation character, which is **already** connected to the standard character rig.

Each mass object can be connected to only **one** rig node, and vice versa. Usually, there will be more mass objects in your reference character than the number of rig nodes in the standard character rig. You will need to decide on a case-by-case basis which mass object is most appropriately driven by which rig node. Keep in mind that you will only need to set up these connections once.



To automatically connect mass objects to rig nodes:

Character Edit Mode now includes an **Auto Create Connections** setting, which is turned on by default.

This means that whenever you **snap** joints together using the Move tool, *endorphin* now **automatically** creates a new mass object connection. This feature is very useful, and means you will rarely need to manually create a mass object connection. This feature was first introduced in *endorphin* 2.6. Prior to *endorphin* 2.6, all connections between mass objects and rig nodes were created manually.

For example, suppose you use the **Move** tool to snap the **LeftShoulderJoint** of the simulation character to the corresponding left shoulder joint of the reference character. Most of the time, this means you will also want to connect the **LeftUpperArmMass** mass object of the simulation character to the corresponding left upper arm mass object of the reference character via the **LeftUpperArm** rig node of the Standard Character Rig.

When using **Auto Create Connnections**, keep the following points in mind:

- If the **Auto Create Connections** setting is turned on, *endorphin* will automatically create mass object connections whenever joints from the simulation character are snapped joints on the reference character, or vice versa. This setting is turned on by default whenever you start *endorphin*. You should only turn this setting off whenever you need to **override** the automatically-created connections.
- If **Auto Create Connections** is on, any existing mass object connection—even a manually created one—will be **replaced** if you snap the corresponding joint to a different joint. Also, keep in mind that if **Auto Create Connections** is on, any existing mass object connection—even a manually created one—will be **removed** if you manually separate a joint from a joint that it has been previously snapped to.
- If **Auto Create Connections** is on, connections between mass objects and rig nodes are mirrored if the corresponding mass objects are mirrored by using one of the Mirror tools.
- The **Auto Create Connections** setting is only available if you are editing a simulation-character pair in Character Edit Mode. If you are editing a simulation scharacter alone, or if you are editing a prop character, this setting is not available.

To manually connect mass objects to rig nodes:

1. **Open** the simulation-reference character pair in Character Edit Mode.
2. Select **View > Motion Transfer Editor** to display the **Motion Transfer Editor**. The keyboard shortcut is **M**.
3. Click the **Activate Reference Character** button in the Main Toolbar to activate the reference character.
4. Turn off the **Auto Create Connections** setting. This allows you to make manual changes to the mass object connections, and ensures that your changes are not overridden when editing the character using the viewport.
5. The **Rig Nodes** list contains three columns. The **left** column contains the list of rig nodes in the standard character rig. The **central** column contains the list of corresponding mass objects in the reference character. It will initially be empty for new reference characters. The **right** column contains the list of corresponding mass objects in the simulation character.

   The **Mass Objects** list contains an indented list of mass objects in the reference character. There is one mass object per joint in the reference character. The indentations reflect the corresponding joint hierarchy. The name of each mass object is the same as the name of its corresponding parent joint, with a **–Mass** suffix.

   - To **connect** a rig node and a mass object, select the rig node from the Rig Nodes list and the mass object from the Mass Objects list, and click **Connect**. When a mass object has been connected to a rig node, the mass object is displayed in bold.

- To **disconnect** a rig node from a mass object, select the rig node from the Rig Nodes list, or the mass object from the Mass Objects list, and click **Disconnect**.

6. When you have finished connecting rig nodes to mass objects, click **OK**.

Rig node strengths

Each rig node has corresponding position and angular **strength** values that specify the degree to which the motion of the source mass object drives the motion of the target mass object.

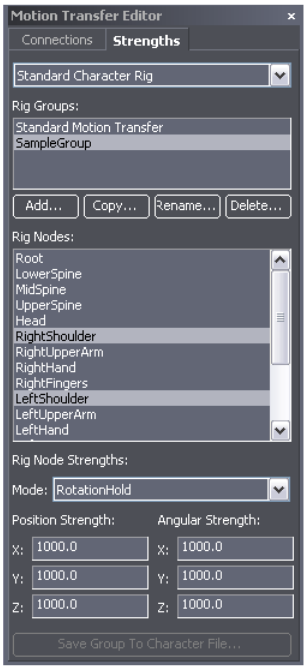
There are three **positional** strength values to specify how strongly motion is driven in each of the three local X, Y and Z directions. There are also three **angular** strength values to specify how strongly rotations are driven about each of the three local X, Y and Z directions.

You can create different sets of rig node strengths. Each set is called a rig node **group**. For example, you may have a rig node group in which all rig nodes are very strong, and another rig group in which the rig node strengths for the right arm have been reduced.

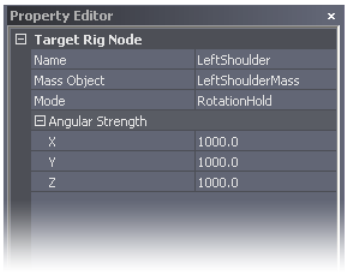
All characters have a rig group called **Standard Motion Transfer**. This group cannot be edited or deleted, and contains the default rig node strengths for standard motion transfer.

Characters also have a rig group called **Advanced Motion Transfer**. A key feature of this group is that it uses **global** X,Y and Z directions for its translational strengths. This rig group is not yet extensively used in the current release of *endorphin*. However, it will be more fully described and utilized in subsequent releases.

Rig node strength in motion transfer editor



Rig node strength in Property editor



Editing simulation and reference characters

You can edit rig node strengths for both simulation and reference characters.

- Most of the time you will be editing **simulation** character rig node strengths. These are used when you are **importing** animation onto the simulation character.
- Occasionally you may also edit **reference** character rig node strengths. These are used when you are **exporting** animation back onto the reference character.

Editing character definitions and instances

You can edit rig node strengths in character definitions, and also in instances of characters in scenes.

- Each character has its own set of rig node strengths stored in its character **definition**. These settings are stored in the character **.nmc** file, and are the default settings used whenever you add an instance of that character to a scene. You can edit these strengths when you are editing the character in Character Edit Mode.

- You can also edit rig node strengths for instances of characters in **scenes**. This modifies the local instance of the character in the scene, but does **not** modify the original character definition.

Adding and removing rig groups

Rig groups contain a set of rig nodes and rig node strengths. The default rig group is the **Standard Motion Transfer** rig group. You cannot edit, delete or rename this group, and you cannot modify any of the rig node strengths in this group.

However, you create new **rig groups**, and edit the rig node strengths in these custom groups. Each rig group contains a copy of the rig nodes in the Standard Character Rig or Advanced Character Rig.

To add or remove rig groups:

1. Select **View > Motion Transfer Editor** to display the **Motion Transfer Editor**. The keyboard shortcut is **M**.
2. If you are editing a character, or pair of characters, in Character Edit Mode, the Motion Transfer Editor will display the settings for the **active** character.

   If you are editing a **scene**, you must select the character you want to edit. You can edit any character, other than the Environment character. A useful feature of the Motion Transfer Editor is that the character selection is **sticky**. This means that once you have specified a character by selecting it, you can deselect that character without clearing the settings in the Motion Transfer Editor.
3. Select **Standard Character Rig** from the Rig Type drop-down box.
4. The **Rig Groups** list contains the list of rig node groups for this character. There will always be one group called Standard Motion Transfer. Select the rig group that you want to edit.

   - To add a new rig group, click **Add…**. Specify a name for the rig group and click **OK**. All rig groups must have unique names. The default rig node settings for this new group will be copied from the Standard Motion Transfer rig group.

- To copy a rig group, click **Copy…**. Specify a name for the rig group and click **OK**. All rig groups must have unique names. The default rig node settings for this new group will be copied from the selected rig group.
- To delete an existing rig group, select the rig group and click **Delete…**. A confirmation dialog will appear asking if you want to delete the rig group. Click **OK** to delete the group. You cannot delete the Standard Motion Transfer group. Hold down the **Ctrl** key to select and delete **multiple** groups.
- To rename an existing rig group, select the rig group and click **Rename…**. The **Rename Rig Node Group** dialog appears. Specify a new name, and click **OK** to rename the group. All rig groups must have unique names. You cannot rename the Standard Motion Transfer group. You can also **double-click** an existing rig group to display the Rename Rig Node Group dialog.

Editing rig node strengths

Each rig node has corresponding position and angular strength values that specify the degree to which the motion of the source mass object drives the motion of the target mass object.

You can edit rig node strengths for character definitions in Character Edit Mode. Alternatively, you can edit rig node strengths for instances of characters in scenes.

You cannot edit the rig node strengths in the **Standard Motion Transfer** rig group. Rather, you create new rig node groups, and edit the rig node strengths in these custom groups. Each rig group contains a copy of the rig nodes in the Standard Character Rig or Advanced Character Rig.

To change rig node strengths:

1. Select **View > Motion Transfer Editor** to display the **Motion Transfer Editor**. The keyboard shortcut is **M**.
2. If you are editing a character, or pair of characters, in Character Edit Mode, the Motion Transfer Editor will display the settings for the **active** character.

   If you are editing a **scene**, you must select the character you want to edit. You can edit any character, other than the Environment character. A useful feature of the Motion Transfer Editor is that the character selection is **sticky**. This means that once you have specified a character by selecting it, you can deselect that character without clearing the settings in the Motion Transfer Editor.
3. Select **Standard Character Rig** from the Rig Type drop-down box.
4. The **Rig Node Settings** list contains the list of rig nodes for the current rig type.

   Select a rig group to display the rig nodes for that group. Then select a rig node to edit its rig node settings. You can select **multiple** rig groups by holding down the **Ctrl** key while you select rig groups. Similarly, you can can select **multiple** rig nodes by holding down the **Ctrl** key while you select rig nodes.
5. Each rig node has three position strengths and three angular strengths. These define strengths for motion along, and rotation about, the local X, Y and Z directions of the corresponding mass object. Strength is a dimensionless parameter with a default value of 1000.0, which implies a relatively strong hold. Use smaller values for weaker holds, and larger values for stronger holds. Use a zero strength value for zero hold.

   Each rig node also has a **mode**. You can set the mode to be any of the following values:

- **FullHold** drives positions and rotations of the target mass object with unlimited strength.
- **FullTranslationHold** drives positions of the target mass object with unlimited strength. It does not drive rotations.
- **FullRotationHold** drives rotations of the target mass object with unlimited strength. It does not drive positions.
- **NoHold** does not drive positions or rotations of the target mass object. This is equivalent to disabling the rig node.
- **TranslationHold** drives positions, but not rotations, of the target mass object. However, unlike FullTranslationHold mode, TranslationHold mode has an upper limit on its strength that you specify in the position strength property.
- **RotationHold** drives rotations, but not positions, of the target mass object. However, unlike FullRotationHold mode, RotationHold mode has an upper limit on its strength that you specify in the rotation strength property.
- **TranslationAndRotationHold** drives positions and rotations of the target mass object. However, unlike FullHold mode, TranslationAndRotationHold mode has upper limits on its strength that you specify in the position and rotation strength properties.

6. When you have finished editing rig node strengths and groups, click **OK**.
    - If you are editing a **character** in Character Edit Mode, the rig node strengths will be stored into the corresponding character **.nmc** file when you save the character. If you add an instance of this character to a scene, that character will contain the new rig node strength values. Keep in mind that any existing scenes containing instances of this character will **not** be updated.
    - If you are editing a **scene**, the rig node strengths will be stored into the scene **.ens** file when you save the scene. You will not affect the original character **.nmc** file.

Copying rig node strengths into character definitions

When you are editing the rig node strengths of characters in a **scene**, you may want to save these settings into the corresponding character definition files, so that you can use these settings in conjunction with dynamic motion transfer in other scenes.

This is a common workflow when you are fine-tuning rig node strengths in one particular scene, and then want to have them available in the character definition so that you can use them for importing and exporting animation data using dynamic motion transfer.

To copy rig node strengths from a scene to a character definition:

1. Select **View > Motion Transfer Editor** to display the **Motion Transfer Editor**. The keyboard shortcut is **M**.
2. If you are editing a character, or pair of characters, in Character Edit Mode, the Motion Transfer Editor will display the settings for the **active** character.

   If you are editing a **scene**, you must select the character you want to edit. You can edit any character, other than the Environment character. A useful feature of the Motion Transfer Editor is that the character selection is **sticky**. This means that once you have specified a character by selecting it, you can deselect that character without clearing the settings in the Motion Transfer Editor.
3. Select **Standard Character Rig** from the Rig Type drop-down box.
4. Click the **Save Group To Character File…** button in the **Motion Transfer Editor**. The currently selected rig node strength group will be copied into the character definition **.nmc** file of the currently selected character. If a group with this name already exists in the character file, the current settings will replace those settings in the character file.

   You can copy **multiple** rig groups into the character definition file. Hold down the **Ctrl** key while selecting the rig groups that you want to copy, and this click the **Save Group To Character File…** button.

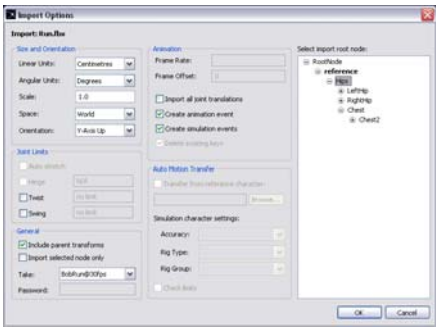
   - This function is **not** available when the Standard Motion Transfer group is selected, since every character has a copy of these settings anyway.
   - This function is **not** available when the selected character is a standard simulation character. It is only available when the selected character is

a custom simulation character or a reference character. This is because the **Standard Simulation Character.nmc** is not available for editing.

- By definition, this function is available when editing scenes. It is not available in Character Edit Mode.

Importing animation data

You can import animation data onto *endorphin* characters using a variety of standard animation formats.



Import file formats for animation data

- **FBX** is a common format for importing animation data. You can import both binary and text FBX file formats. FBX is commonly used when you are working with Maya.
- **XSI** (also known as dotXSI) is another common format for importing animation data, and is commonly used when working with SOFTIMAGE|XSI.
- **BVH** is a lightweight text file format that contains both skeletal and animation data, and is commonly used when working with 3dsMax.
- **AMC** is the Acclaim Motion Capture animation format, and contains animation data. You will commonly use AMC animation data when working with characters created by importing ASF (Acclaim Skeletal Format) character definition files.
- **V** files are the Vicon Motion Capture system animation data files. You will commonly use V files when working with characters created by importing Vicon VST and VSK character definition files.

To import animation data:

1. **Select** a character that you want to import animation onto.

- If you select a **simulation** character, you must have also defined a corresponding **reference** character. The animation data that you want to import must match the character joint hierarchy of this reference character. You will usually need to use dynamic motion transfer to transfer the data from the source reference character to the target simulation character.

  However, if you are importing animation that you have generated within *endorphin* itself using the **same** simulation character that you are importing onto, you will **not** need to use dynamic motion transfer. You will be able to import the motion **directly**. This is because the animation will already be defined for this character.

  (If you are importing motion onto a simulation character that was generated in endorphin using a **differently-shaped** simulation character, then you will still need to use dynamic motion transfer. This is because the source and target characters will have different joint lengths, even though their joint hierarchies themselves will match in terms of topology and naming conventions.)

- If you select a **prop** character, the animation data that you want to import must match the character joint hierarchy of this prop character.

- If you do not select any character, a new prop character will be created in the scene, and the animation added to this character. However, we generally do not recommend this workflow.

2. Select **File > Import…** to display the **Select File To Import** dialog. The keyboard shortcut is **I**. You can also click the **Import** button  in the Main Toolbar.

   Alternatively, **right-click** on a character timeline and select **Create Animation Event**. This will also display the Select File To Import dialog. You do not need to have a character selected if you use this method.

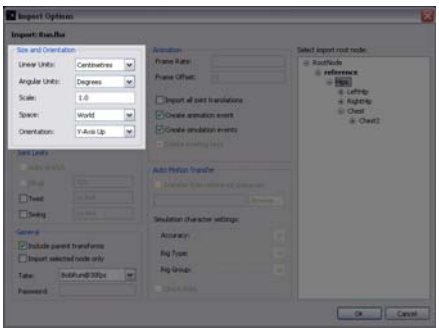
3. Select a file containing animation to import. You can choose any FBX, XSI, BVH, AMC or Vicon V file. The only requirement is that the file must match the character joint hierarchy of the selected character (or its corresponding reference character in the case of simulation characters).

4. The **Import Options** dialog is displayed. This dialog contains a number of options that control how the data is imported. Many of these options are only relevant when creating new characters by importing animation data. When you are importing data onto an existing character, many of these options are not used.

   Specify the desired options and click **OK** to import the animation onto the selected character. If you have turned on Create Animation Event, a new animation event will be created. Otherwise, a series of explicit keyframes will be created. If no character was currently selected, a new character will have been created, and the animation applied to that character.

Import options: Size and orientation

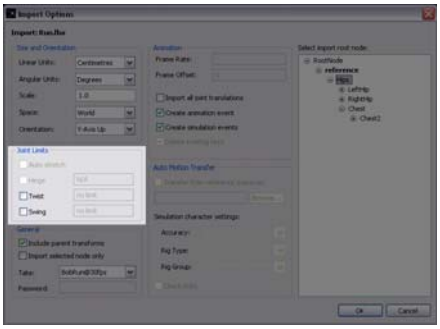
The Size and Orientation options let you specify the linear units, angular units and scale to apply when you are creating a new character from animation data, as well as the angular units and space to apply when you are importing animation onto an existing character.



- **Linear units** specify how positions are stored in the animation data. Options are metres, centimetres, millimetres, feet and inches. For example, default units for Maya are centimetres. Only used when creating new characters.
- **Angular units** specify how rotations are stored in the animation data. Options are degrees, radians and revolutions. Most animation files store angles as degrees. Used when creating new characters or importing onto existing characters.
- **Scale** specifies an additional scale factor to apply when importing the character hierarchy. Default value is 1.0. Only used when creating new characters.
- **Space** specifies whether to import animation in world space or character space. Default value is world space. Use character space if you have edited the location of the character cube of the target character. Only used when importing onto existing characters.
- **Orientation** specifies whether the animation data assumes that the y-axis is the vertical direction or the z-axis is the vertical direction. *endorphin* assumes the y-axis is vertical direction. Maya assumes y-axis is vertical direction; 3ds Max assumes z-direction is vertical direction. Used when creating new characters or importing onto existing characters.

Import options: Joint limits

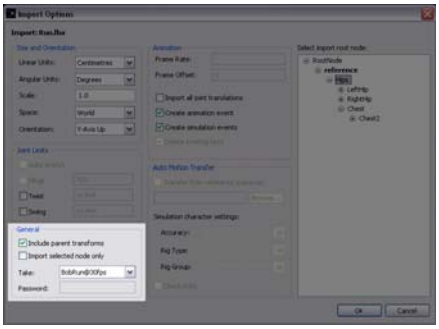
The Joint Limit options let you specify the default joint limit ranges for new skeletal joints. These options only apply when you are creating a new character from animation data.



- **Auto stretch** is currently unused.
- **Hinge joint limit** is currently unused.
- **Twist joint limit** specifies the default twist joint limit range for skeletal joints. If this option is turned on, you can specify the default twist joint limit in the text box. The default twist limit is 45.0 degrees. If this option is turned off, there is no twist limit applied when creating new skeletal joints. This option is useful for rapidly generating a character with reasonable joint limits. Specific joint limits can also be set when editing the character in Character Edit Mode. Only used when creating new characters.
- **Swing joint limit** specifies the default swing joint limit range for skeletal joints. If this option is turned on, you can specify the default swing joint limit in the text box. The default swing limit is 45.0 degrees. If this option is turned off, there is no swing limit applied when creating new skeletal joints. This option is useful for rapidly generating a character with reasonable joint limits. Specific joint limits can be also set when editing the character in Character Edit Mode. Only used when creating new characters.

Import options: General

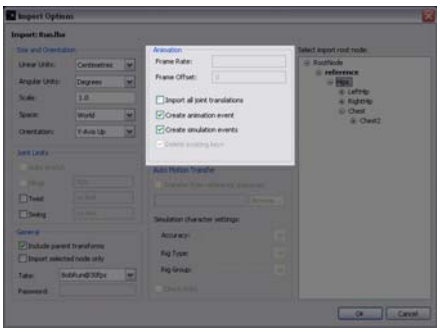
The General options include some settings that are used in conjunction with the import root node, as well as other general-purpose settings.



- **Include parent transforms** is used in conjunction with the import root node. See the **Select root node options** for more details.
- **Import selected node only** is used in conjunction with the import root node. See the **Select root node options** for more details.
- **Take** specifies which take of animation data to import. The default is Take 001. Only available when importing FBX files that have been generated in MotionBuilder.
- **Password** specifies the password to use for password-protected files. Only available when importing FBX files.

Import options: Animation

The Animation options let you specify whether or not to create animation events, whether or not to create simulation events, and control other settings such as frame rates and frame offsets.

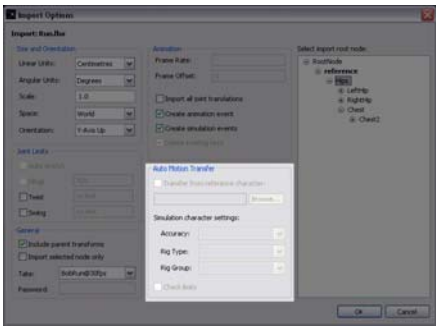


- **Import all joint translations** specifies whether or not to consider animated joint translations when importing animation data. By default, only joint rotations are imported. If you have animation data that contains joint translations that you want to preserve—these could be subtle joint compressions captured by motion capture, for example—you can turn this setting on.
- **Create animation event** specifies whether to create a new animation event to store the animation keyframe data. If this option is set, a new animation event marker is created. If this option is not set, a series of individual keyframes are created directly. By default, this option is turned on. Creating animation events is almost always preferable to creating individual keyframes.
- **Create simulation events** specifies whether simulation events should be created at the start and end of the animation event. By default, this option is turned on. When you use animation events, you will usually enclose them with simulation events so that the character is unaffected by the simulation for the duration of the animation event.
- **Delete existing keys** specifies whether to delete any existing keyframes from a character timeline. Only available if Create Animation Event is turned off. By default, this option is turned off.
- **Frame offset** specifies the start frame of the first individual keyframe. This setting only applies if keyframes are being created, rather than animation events.
- **Frame rate** specifies the number of frames per second to import at. Only available for AMC files. For other file formats, the frame rate is contained as part of the animation data itself, and cannot be overridden.

Import options: Auto motion transfer

Auto motion transfer is a key part of the *endorphin* animation pipeline. It is the process of using **dynamic motion transfer** to import reference animation directly onto corresponding *endorphin* simulation characters.

To use auto motion transfer, you first need to create a **simulation-reference character pair**, and then populate the scene with the simulation character. You can then import animation based on the reference character directly onto the simulation character using auto motion transfer. By using different **simulation character rigs**, you can achieve different effects.



Auto motion transfer options

The following import options are available in the Import Options dialog when you are importing animation onto an existing simulation character.

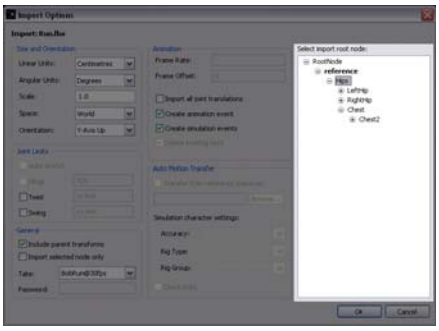
- **Transfer from reference character** specifies whether a reference character is to be used to supply animation that is to be dynamically transferred to the selected simulation character. By default, this option is turned off. If you are importing animation onto a simulation character you will usually need to turn this option on.
- The name and path of the reference character to be used with the dynamic motion transfer needs to be specified. Only used when the **Transfer from reference character** setting is turned on. Click the **Browse…** button to browse for a reference character that corresponds to the selected simulation character. If a corresponding reference character exists, this will be the default reference character, but you can override this. For example, if your selected character is an instance of **Goblin.nmc**, the default reference character will be Goblin_ref.nmc, provided that this file can be found.
- **Accuracy** specifies how precisely motion should be transferred from the source reference character to the target simulation character. The default accuracy is Medium. Accuracy can be Fast, Medium or Accurate.

- **Rig type** specifies which simulation character rig to use for dynamic motion transfer. In *endorphin 2.6*, you are likely to only ever use the Standard Character Rig. In future versions of *endorphin*, other character rigs such as the Advanced Character Rig will also be available.
- **Rig group** specifies which simulation rig group to use for the dynamic motion transfer. The default rig group is the Standard Motion Transfer group. The list of rig groups will be populated by the rig groups stored in the instance of the target simulation character in the scene. Initially, the list of rig node groups will have been defined by the character definition **.nmc** file, but you can also edit character rig nodes in the scene.
- **Check limits** is an advanced option that specifies whether to check that the reference character joint limits fall within the extents of the animation data. Rarely used in most cases and is not fully supported. If this option is turned on, any joint motion which falls outside the allowable range defined by the joint limits is reported in the **Output** window.

Import options: Root node

The **root node** hierarchy displays the node hierarchy in the selected animation file. You can select one of the nodes to specify which part of the hierarchy you want to import. If you are creating a new character, this node specifies the subset of the character that you want to create. If you are importing onto an existing character, this node specifies which joints you want to animate.

To import an entire character, locate the character's root node in the node hierarchy. Sometimes, this node will be some levels down from the root node of the file.



Some of the General options apply to the selected root node for import:

- **Include parent transforms**. This option is only available when importing FBX files. When this option is turned off, the selected node in the node tree is aligned to the origin of the virtual world. When this option is turned on, the selected node is animated relative to the root of the node tree, which is placed at origin of the virtual world. This option is turned off by default.

  This option is useful if you want to import the motion of a single child node, or a set of child nodes, and have the nodes move as though the entire node hierarchy had been imported. For example, if you import the arm of a character during a run cycle, the arm joints will move across the scene as though the entire character had been imported. If this option is off, the shoulder of the arm would remain fixed at the scene origin.

- **Import selected node only**. This option is only available when importing FBX files. When this option is turned off, the selected node and all its child nodes are imported. When this option is turned on, only the selected node is imported. This option is turned off by default.

  This option is useful if you want to import the motion of a single node, and use this motion in conjunction with other characters in a scene using constraint events.

Importing animation data to create characters

You can import skeletal data files to create reference characters in **Character Edit Mode**.

To create new characters, you need to turn on the **Create Reference Character Using** setting in the New Character. You can then browse for a file that contains a character hierarchy. If you create a reference character **without** a corresponding simulation character, we usually refer to the reference character created in this way as a **prop character**.

Import file formats for skeletal data

- **FBX** is a common format for creating characters. It contains both skeletal and animation data; to create characters only the skeletal data is used. You can import both binary and text FBX file formats. FBX is commonly used when you are working with Maya.
- **XSI** (also known as dotXSI) is another common format for creating characters. It contains both skeletal and animation data; to create characters only the skeletal data is used. XSI is commonly used when working with SOFTIMAGE|XSI.
- **BVH** is a lightweight text file format that contains both skeletal and animation data. To create characters, only the skeletal data is used. BVH is commonly used when working with 3dsMax.
- **ASF** is the Acclaim Skeletal Format character definition format. You will commonly use ASF character files in conjunction with importing and exporting animation data using the **AMC** (Acclaim Motion Capture) animation file format.
- **VST** and **VSK** files are the Vicon Skeletal Template and Vicon Skeletal Kinematic file formats containing character skeletal data. You will commonly use VST and VSK character files in conjunction with importing and exporting animation data using the Vicon **V** animation file format.

Import Options used when creating characters

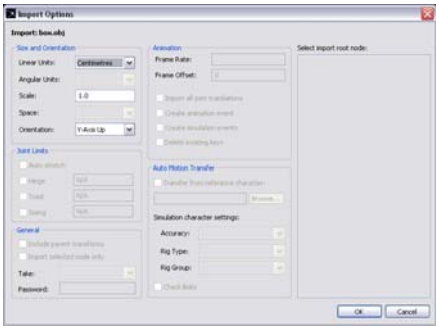
When you import data stored in FBX, XSI, BVH, ASF, VST or VSK files to create new characters, the **Import Options** dialog appears, once you have selected the desired import file. Many of these options are disabled, as they are not relevant when importing character skeletal data, rather than animation data.

- **Size and orientation** and **General** options, as well as the **Import Root Node** setting, are used in the same manner as for animation import.
- **Joint limits**, **Animation** and **Auto Motion Transfer** options are unused when creating characters.

Importing OBJ mesh data to create graphical objects

You can import OBJ mesh data files to create graphical objects in *endorphin*. **OBJ** is a common format for defining polygonal surface meshes. You can import OBJ meshes to create **graphical objects** in Character Edit Mode, and also when editing scenes.

Note New graphical objects are created with a name that corresponds to the OBJ filename. You cannot edit graphical object names using the Property Editor.



Import Options used when creating graphical objects

When you import skin mesh data stored in OBJ files to create new **graphical objects**, the Import Options dialog appears with most options disabled.

- **Linear Units**, **Scale** and **Orientation** options are used in the same manner as for animation import. All other options are disabled and not relevant for OBJ import.

To specify the graphical object parent:

- If you are editing a scene, **select** the character that will own the graphical object before importing the OBJ file. If no character is selected, the graphical object will be created as a child of the Environment character.
- If you are editing a simulation-reference character pair in **Character Edit Mode**, **activate** the character that will own the new graphical object before importing the OBJ file.

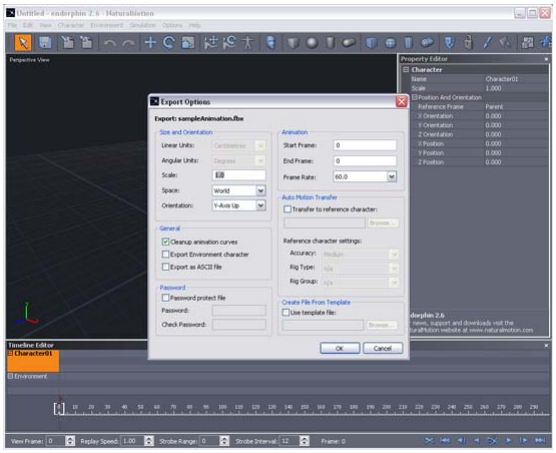
Introducing animation export

Once you have generated motion using *endorphin*, you can **export** this motion into a variety of standard animation formats. Animation data can be exported as FBX, XSI, BVH, AMC or Vicon V files. You can then import this animation into your target animation system—such as Maya, 3dsMAX or SOFTIMAGE|XSI—to complete the animation pipeline.

You can export the motion of **prop** characters and the **Environment** character, as well as the motion of **simulation** characters. Keep in mind that when you export the motion of simulation characters, you will usually use Auto Motion Transfer to transfer the motion from the simulation character to its corresponding reference character.

You can also save the frames of animation as a single **AVI** movie file, or as a series of **bitmap** files.

Note that you **cannot** export animation when using *endorphin* Learning Edition. This is the only functionality that has been removed in *endorphin* Learning Edition.



Exporting animation data

You can export animation data from *endorphin* characters using a variety of standard animation formats.

Export file formats

- **FBX** is a common format for exporting animation data. You can export both binary and text FBX file formats, and you can export both version 5 and version 6 FBX files. FBX is commonly used when you are working with Maya.
- **XSI** (also known as dotXSI) is another common format for exporting animation data, and is commonly used when working with SOFTIMAGE|XSI.
- **BVH** is a lightweight text file format that contains both skeletal and animation data, and is commonly used when working with 3dsMax.
- **AMC** is the Acclaim Motion Capture animation format, and contains animation data. You will commonly export AMC animation data when working with characters defined in ASF (Acclaim Skeletal Format) character definition files.
- **V** files are the Vicon Motion Capture system animation data files. You will commonly export V file data when working with characters defined in Vicon VST and VSK character definition files.

To export animation data:

1. **Select** a character that you want to export animation from.
    - If you select a **simulation** character, you must have also defined a corresponding **reference** character. The animation data that you want to import must match the character joint hierarchy of this reference character. You will need to use dynamic motion transfer to transfer the data from the source reference character to the target simulation character.

- You can also select a **prop** character or the **Environment** character.
- If you want to export motion for **all** the characters, do not select any of the characters.

2. Select **File > Export…** to display the **Export As** dialog. The keyboard shortcut is **Ctrl+E**. You can also click the **Export** button  in the Main Toolbar. If you want to export the motion for all the characters, select **File > Export All…**. This command will export the motion for all characters in the scene, except for the Environment character.
3. Specify the export filename. You can specify a new filename, or you can replace an existing file. Keep in mind that the export file format is based on the **file extension** of the exported file. You should use the **Save as type** combobox in the Export As dialog to specify the file extension, rather than typing it by hand.

   Note If you are exporting FBX files, you must be choose between the **FBX 5.0** and **FBX 6.0** file formats by selecting the desired format from the **Save as type** list.
5. The **Export Options** dialog is displayed. This dialog contains a number of options that control how the data is exported. Specify the desired options and click **OK** to export the animation from the selected character.

   If you are exporting motion for **all** characters, *endorphin* will generate a unique file for each character in the scene. Each file will have a unique number added to the end of the filename to ensure filename uniqueness—for example, **FightScene1.fbx**, **FightScene2.fbx**, **FightScene3.fbx** and so on.

   Note If you export all characters to an FBX file format, a dialog appears asking if you would like to use a single file for export. If you select **Yes**, the motion for all characters is added to a single FBX file. If you select **No**, a separate FBX file is created for each character. If you select **Cancel**, the export process is cancelled.

Exporting marker motion

Markers are small spherical graphical items that are associated with mass objects. By default, markers are not displayed. You can show or hide markers using the viewport display filter. The motion of markers is exported when animation data is exported using the **CSM** (Character Studio Marker) file format. Keep in mind that this pipeline is not widely used and is no longer actively supported.

Exporting movie files

After you have generated animation, you can save this animation out to an **AVI movie** file.

To save your animation as an AVI movie file:

1. **Resize** the active viewport to reflect the width and height of each frame of the AVI file. Keep in mind that generating AVI files using the full screen width and height can be quite slow. The AVI export process cannot be manually stopped once it has started, so it is important not to choose a viewport size that makes AVI generation excessively slow.
2. Adjust the viewport **camera settings**, such as the viewport display filter and camera angle. It is good practice to scrub through the timeline to ensure that the AVI file will contain all the desired motion.
3. Select **File > Export Movie…** and browse to specify an export path and filename.
4. The **Export Options** dialog appears with most options disabled.
5. Specify the **Start Frame** and **End Frame** to be rendered, and the desired **Frame Rate** of the AVI file. You can choose the frame rate from a list of industry-standard frame rates.
6. Click **OK**. *endorphin* will generate the AVI file. You will notice the viewport playing back the animation from the framebuffer while the AVI file is being generated.

   Note No dialog appears when the AVI file has been generated. When the viewport finishes playing back the animation, this is an indication that the AVI file has been generated.

Exporting bitmap files

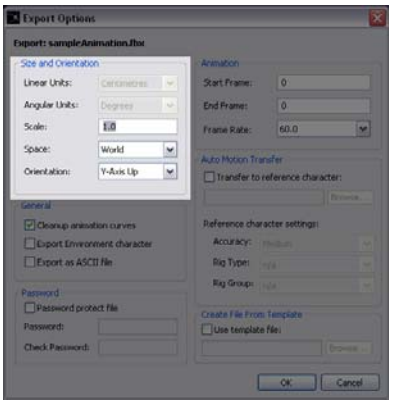
When you are generating animation, you can optionally choose to save each frame of the viewport as a separate **bitmap** file. Unlike AVI export, bitmap generation occurs **during** a simulation.

To save your animation as a set of bitmap files:

1. Adjust the viewport **camera settings**, such as the viewport display filter and camera angle. It is good practice to scrub through the timeline to ensure that the bitmap files will contain all the desired motion.
2. Deselect all objects in the scene. This will ensure that the **Timeline Settings** are displayed in the Property Editor.
3. Locate the **Bitmap Export** properties, and turn on the **Enabled** setting.
4. Adjust the other settings as required:
    - The **Path** specifies the output folder. The default path is **bin\NMVids**, relative to the installation folder.
    - The **Horizontal Resolution** and **Vertical Resolution** specify the size of each bitmap. The default bitmap size is 640 x 480 pixels.
    - The **Frame Count** specifies the maximum number of bitmaps to generate. The default frame count is 200 frames.
    - The **Frame Rate** specifies the number of frames per second to generate. The default frame rate is 25 frames per second.
    - The **Render View Only** setting specifies what data is to be rendered into each bitmap. If this setting is turned on, only the viewports are rendered as bitmaps. If this setting is turned off, the entire application window is rendered along with the viewports. This setting is turned on by default.

Export options: Size and orientation

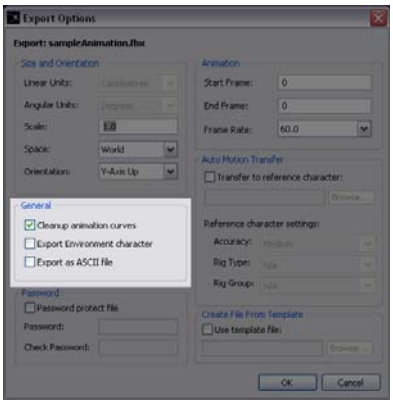
The Size and Orientation options let you specify the linear units, angular units and scale to apply when you exporting animation data.



- **Linear units** specify how positions are to be stored in the animation data. Options are metres, centimetres, millimetres, feet and inches. For example, default units for Maya are centimetres. Not available for all export formats.
- **Angular units** specify how rotations are to be stored in the animation data. Options are degrees, radians and revolutions. Most animation files store angles as degrees. Not available for all export formats.
- **Scale** specifies an additional scale factor to apply when exporting the character hierarchy. Default value is 1.0.
- **Space** specifies whether to export the translation and rotation of the character root joint in terms of the world coordinate system or the character's local coordinate system. Default value is World space.
- **Orientation** specifies whether the animation data will assume that the y-axis is the vertical direction or the z-axis is the vertical direction. *endorphin* assumes the y-axis is vertical direction. Maya assumes y-axis is vertical direction; 3ds Max assumes z-direction is vertical direction. Used when creating new characters or importing onto existing characters.

Export options: General

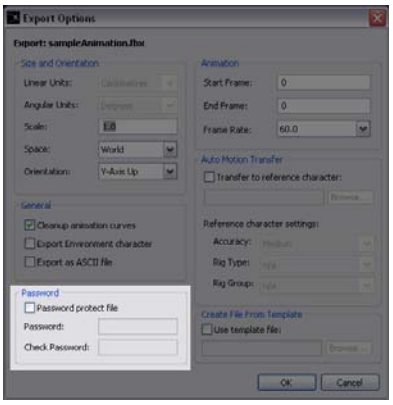
The General options include various general-purpose settings.



- **Cleanup animation curves**. When this setting is on, exported rotation curves are processed to ensure that there are no 360 degree discontinuities. For example, if a joint rotation at a given frame is 359 degrees, and at the subsequent frame it is 1 degree, the cleanup tool will convert the second angle from 1 degree to 361 degrees to ensure a smoother rotation curve. This setting is on by default.
- **Export Environment character** is only available when exporting using the **FBX** file format. This setting specifies whether to export the Environment character. If this setting is turned on, and you are exporting to a **single** file, the Environment character will be exported into this file, along with any other specified characters. This setting is off by default.
- **Export as ASCII file** is only available when exporting using the FBX file format. When this setting is on, FBX **text** files are generated. When this setting is off, FBX **binary** files are generated. This setting is off by default.

Export options: Password

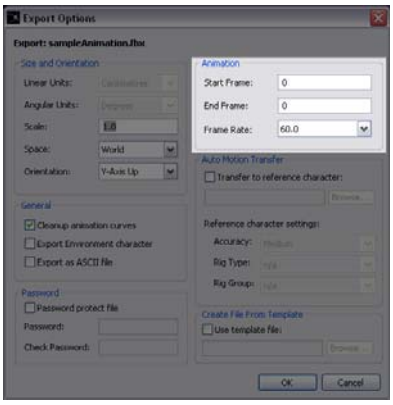
The Password options allow you to password-protect files. These options are only available when exporting using the FBX file format.



- **Password protect file** specifies whether or not to password-protect the exported FBX file. This setting is off by default.
- **Password** and **Check password** are available when you turn on the **protect password** setting. You must specify a password in the Password field, and then retype it correctly into the Check password field. You cannot copy-and-paste from one field to the other.

Export options: Animation

The Animation options let you specify the range of frames to export, and the frame rate.

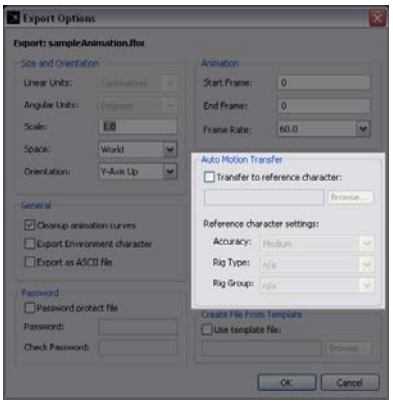


- **Start frame** specifies the first frame of data in the frame buffer to export. The default start frame will correspond to the Timeline Editor **Save Range** start frame, but you can override this. The start frame should not be greater than the total number of frames stored in the framebuffer.
- **End frame** specifies the last frame of data in the frame buffer to export. The default end frame will correspond to the Timeline Editor **Save Range** end frame, but you can override this. The end frame should not be greater than the total number of frames stored in the framebuffer.
- **Frame rate** specifies the frame rate of the exported animation. The frame rate only applies when exporting using the **FBX** file format. For FBX format, you can choose from the following frame rates: 120, 60, 50, 30, 29.97, 25 and 24 frames per second.

Export options: Auto motion transfer

Auto motion transfer is a key part of the *endorphin* animation pipeline. It is the process of using **dynamic motion transfer** to export animation from *endorphin* simulation characters onto corresponding reference characters.

To use auto motion transfer, you first need to create a **simulation-reference character pair**, and then populate the scene with the simulation character. After simulating, you can then export animation from the simulation characters onto their corresponding simulation characters using auto motion transfer. By using different **reference character rigs**, you can achieve different effects.



Auto motion transfer options

The following export options are available in the Export Options dialog when you are exporting animation from a simulation character.

- **Transfer to reference character** specifies whether a reference character is to be used as a target for dynamically transferred motion from the selected simulation character. By default, this option is turned off. If you are exporting animation from a simulation character you will usually need to turn this option on.
- The name and path of the reference character to be used with the dynamic motion transfer needs to be specified. Only used when the **Transfer to reference character** setting is turned on. Click the **Browse…** button to browse for a reference character that corresponds to the selected simulation character. If a corresponding reference character exists, this will be the default reference character, but you can override this. For example, if your selected character is an instance of **Goblin.nmc**, the default reference character will be Goblin_ref.nmc, provided that this file can be found.

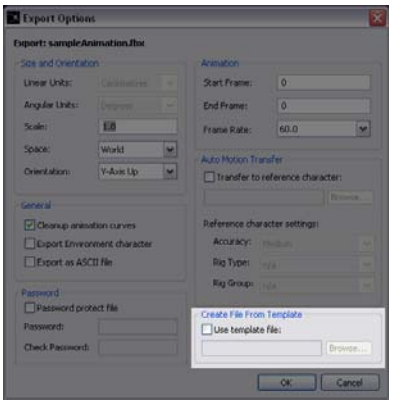
- **Accuracy** specifies how precisely motion should be transferred from the source simulation character to the target reference character. The default accuracy is Medium. Accuracy can be Fast, Medium or Accurate.
- **Rig type** specifies which reference character rig to use for dynamic motion transfer. In *endorphin 2.6*, you are likely to only ever use the Standard Character Rig. In future versions of *endorphin*, other character rigs such as the Advanced Character Rig will also be available.
- **Rig group** specifies which reference rig group to use for the dynamic motion transfer. The default rig group is the Standard Motion Transfer group. The list of rig groups will be populated by the rig groups stored in the instance of the target reference character in the scene. Initially, the list of rig node groups will have been defined by the character definition **.nmc** file, but you can also edit character rig nodes in the scene.

Export options: Create file from template

Create file from template is available when you export data using the FBX and XSI file formats. This option allows you to choose an existing FBX or XSI file to use as a basis, or **template**, for the exported file. The animation data of the selected character is inserted into a **copy** of the specified template file.

The benefit of this approach is that *endorphin* is used to simply add animation data to your file. Although *endorphin* can create entire scenes that include character hierarchies and animation data, you will usually have application-specific character hierarchies that need to be preserved. Using the **Create File From Template** settings is the best way to achieve this.

You can use this option when using the **Export All** command, as well as when using the **Export** command.



- **Use template file** specifies whether to merge with an existing file. This setting is turned off by default. This setting is only available when exporting using the FBX and XSI file formats.
- The name and path of the template file to be used needs to be specified. Only used when the **Use template file** setting is turned on. Click the **Browse…** button to browse for an existing FBX or XSI file to merge with. Keep in mind that the template file is **not** modified. One or more new files are created based on **copies** of the template file, and animation data is inserted into these copied files.

Chapter 21

Using *endorphin* with…

Introducing third-party scripts

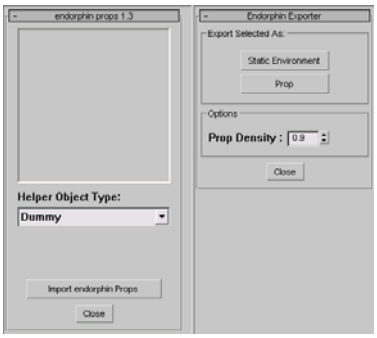
endorphin ships with a range of third-party animation scripts to help improve the animation data pipeline. These scripts are installed in the **Resources\Third Party\Scripts** folder.

Maya MEL scripts

- Maya MEL scripts are installed in the **Resources\Third Party\Scripts\Maya** folder.
- The **endorphin_Exporter.mel** MEL script is used when you want to create prop characters or animated cameras in Maya. This script generates corresponding *endorphin* **.nmc** character files and **.nmcam** animated camera files.
- For more information, read the **endorphin_Exporter_ReadMe.txt** file.

3dsMax scripts

- 3dsMax scripts are installed in the **Resources\Third Party\Scripts\3dsMax** folder.



- The **endorphin_ExporterPropExporter.ms** script is used when you want to create prop characters in 3dsMax. This script generates corresponding *endorphin* **.nmc** character files.
- The **endorphin_CameraExporter.ms** script is used when you want to create animated cameras in 3dsMax. This script generates corresponding *endorphin* **.nmcam** animated camera files.

- The **endorphin_CSMpropImporter.ms** script is used when you want to import CSM (CharacterStudio Marker) data back into 3dsMax. The CSM animation pipeline was introduced in early releases of *endorphin* and is no longer supported.
- For more information, read the **endorphin_MaxScripts.txt** file.

Using the Maya Exporter script

endorphin ships with a Maya MEL script that allows you to create prop characters in Maya and export them as endorphin .nmc prop characters. Currently this script only allows you to create prop characters. In future releases of endorphin, this script may be extended to allow you to create endorphin simulation characters as well.

To create a prop character in Maya:

1. Open the Maya **script editor** window.
2. Select **File > Source script**.
3. Browse to **Resources\Third Party\Scripts\Maya** and load **endorphin_Exporter.mel**.
4. In the *endorphin* exporter window, use the tools provided to create your desired object . See the inline help in the MEL script for details of how to use the tools.
5. When ready, export your prop character.

To use the prop character in *endorphin*:

1. Select **Character > Add Character**.
2. Click the **Browse** button and navigate to the prop character saved from Maya.
3. Click **OK**. The prop character will be added to the scene. You can now add motion to this character, or attach it to other characters, for example.

To modify the prop character in *endorphin*:

1. Select **File > Open Character**.
2. Click the **Browse** button and navigate to the prop character saved from Maya.
3. Click **OK**. You are now in Character Edit Mode. You can make changes to the prop character, such as modifying its mass objects, collision objects and materials.

Introducing the 3dsMax Biped pipeline

You can use *endorphin* with **3dsMax Bipeds** in an animation pipeline.

To do so, you need to configure the pipeline. You can then export animation data from 3dsMax Biped into *endorphin*, and export animation data back from *endorphin* into 3dsMax. This pipeline requires the use of **two** file formats: **FBX** for transferring data from Biped to *endorphin,* and **BVH** for transferring data from *endorphin* back into Biped.

The following steps outline the process of transferring data from 3dsMax to *endorphin* and back.

Step 1: Preparing your 3dsMax Biped character

In this step you will prepare your 3dsMax Biped character so that it can be used in *endorphin*.

You need to ensure the following:

- In **Figure** mode, ensure that your 3dsMax Biped 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.
- The character should be facing along the **positive Z axis** if you are using the Y axis as the vertical direction. Alternatively, the character should be facing along the **negative Y axis** if you are using the Z axis as the vertical direction.
- Ideally, you should set the 3dsMax Display Units to **Meters**, and the 3dsMax System 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 3dsMax

In this step you will export your character as a **FIG** file, and also save it as an **FBX** file.

To save your 3dsMax Biped to FIG and FBX files:

1. Launch 3dsMax and load the Biped character that you want to simulate in *endorphin*.

2. Select the Biped and browse the **Motion Panel**.
3. Place your Biped into **Figure** mode, and ensure that it is a T-pose.
4. If you are going to animate you character from the default origin position in *endorphin*, rather than importing an animation cycle to begin the simulation, then you should move your Biped so that it is standing on the ground at (0.0, 0.0) in 3dsMax before exporting. This is to ensure it your character is not partially inside the ground plane when *endorphin* begins simulating.
5. Save your character as a **FIG** file.
6. While you are in Figure mode, export your Biped as an **FBX** file.

Step 3: Creating corresponding *endorphin* import characters

In this step, you will create a **pair** of characters in *endorphin* that represent your 3dsMax Biped character. Once you have created this simulation-reference character pair, you will be able to use them to import animation from 3dsMax 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 3dsMax Biped 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 FBX file that you saved in Step 2. In the **Import Options** dialog, select the **Bip 01Pelvis** node from the joint hierarchy tree, and then click **OK**.
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 you can specify a 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*.

Modifying mass and collision objects

You can also resize and reshape the **mass objects** and **collision objects** of the simulation character, and also create additional collision objects.

A useful technique is to use an imported OBJ mesh file—representing a skin definition—as a guide while modifying the distribution of mass and collision objects.

See the User Guide for more information on using Character Edit Mode to create custom characters.

Step 4: Creating a corresponding *endorphin* export character

In this step, you will create another *endorphin* reference character. This character will be used when you **export** animation from *endorphin* into 3dsMax using dynamic motion transfer.

Normally, you would be able to use the **same** reference character for both animation import and export. However, the 3dsMax pipeline is slightly more complicated—it requires exporting animation from 3dsMax using **FBX** files, but importing animation back into 3dsMax Bipeds using **BVH** files. Bipeds require a specific BVH skeletal naming convention that differs from the naming convention in their exported FBX file.

In short, we need to create an **additional** *endorphin* reference character based on the standard 3dsMax Biped BVH skeleton. You can create these reference characters by basing them on the **Character Studio BVH Character.nmc** character in **Resources\Third Party\Characters**. This is an *endorphin* character that has been designed with the correct BVH naming convention.

To create an BVH export reference character:

1. Select **File > New Character…**.
2. Select the **simulation** character you created in **Step 3**.
3. Turn on the **Source Reference Character** setting and browse to the **Resources\Third Party\Characters** folder.
4. The character **Character Studio BVH Character** should appear in the character list. Select this character and click OK. *endorphin* will create a simulation-reference character pair using the Character Studio BVH Character as the reference character.
5. Click the **Activate Reference Character** button in the Main Toolbar. This activates the reference character. Align the joints of the reference character to match the joints of the simulation character.

   Keep in mind that this is the **opposite** of the usual Character Edit Mode workflow. The usual workflow is to reshape a simulation character to match a reference character. However, in this case, the simulation character has already been reshaped to match its reference character in Step 3. In this step you are now matching a **new** reference character to the reshaped simulation character.
6. Once the two character skeletons are aligned, select **File > Save Character** and save the character. This will save the characters as separate simulation and reference character files. However, you will only use the reference character for transferring motion from *endorphin* to 3dsMax.

Naming conventions

It is good practice to choose similar names for the characters you create in Step 3 and Step 4, except that you must be able to distinguish the characters used in data import from the characters you use for data export.

For example, if your 3dsMax Biped character is a warrior, you might create **WarriorImport.nmc** and **WarriorImport_ref.nmc** in Step 3, and **WarriorExport.nmc** and **WarriorExport_ref.nmc** in Step 4.

Step 5: Exporting animation from 3dsMax into *endorphin*

In this step, you are going to **export** animation that you have created in 3dsMax, and then **import** it into *endorphin*, using an FBX file to transfer the data.

To export animation from 3dsMax:

1. Exit **Figure Mode**.
2. Load the animation that you want to export to *endorphin*.
3. Select **File > Export** and save the animation as an **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. 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 3dsMax with the corresponding character. Click **Open** to display the **Import Options** dialog.
7. In the Import Option dialog, turn on the **Import From Reference Character** setting.

8. Click the **Browse** button in the Auto Motion Transfer settings. Select the corresponding reference character file you created for importing data. For example, if you are working with the warrior character described in Step 4, you will need to browse to **WarriorImport**, or whatever name you have chosen for this import reference character.
9. In the **General** settings, turn on the **Include parent transforms** setting.
10. In the node hierarchy tree, select the **Bip01Pelvis** 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 3dsMax

In this step, you are going to **export** animation that you have created in *endorphin*, and then **import** it into 3dsMax, using a BVH 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 BVH Files. You can enter a new name, or choose an existing BVH file to replace it. Click **Save** to display the **Export Options** dialog.
4. In the **Export Options** dialog, turn on the **Export To Reference Character** setting.
5. Click the **Browse** button in the Auto Motion Transfer settings. Select the corresponding reference character file you created for exporting data. For example, if you are working with the warrior character described in Step 4, you will need to browse to **WarriorExport**, or whatever name you have chosen for this export reference character.
6. Click **OK** to export the animation to the BVH 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 3dsMax:

1. In 3dsMax, select the Biped character that you want to import animation onto.
2. Browse to the **Motion Panel**.
3. In the **Motion Capture** rollout, click **Load Motion Capture File**.
4. Load the BVH file saved from *endorphin* earlier in this step.
5. Click **OK**.
6. Enter **Figure** mode and load the **FIG** file saved in **Step 2: Exporting your character from 3dsMax**.
7. Exit Figure mode.
8. Press **Play** in 3dsMax to see the animation applied to the Biped. You can now modify this animation like any Biped animation.

Introducing the Lightwave pipeline

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

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

The following steps outline the process of transferring data from Lightwave to *endorphin* and back.

Step 1: Configuring Lightwave

Data transfer between *endorphin* and LightWave requires using the **Lightwave FXB plugin**. This plugin is available from Alias at **www.alias.com/glb/eng/community/downloads.jsp**.

You will need to download the **FBX 6.0.2 Plug-in for Lightwave 8.2**. If you are using a different version of Lightwave, download the corresponding FBX plugin for your version of Lightwave. This plugin comprises **three** utilities: an **FBX export** utility, an **FBX import** utility and an **FBX file merge** utility for merging of FBX files with existing scenes in LightWave.

To install the Lightwave FBX plugin:

1. Copy the downloaded file **fbxlw82.p** into the **Plugins\input-output** folder in your LightWave installation directory.
2. In LightWave, browse to the **Utilities** tab and select **Plugins > Add Plugins**.
3. Navigate to the location of the FBX plugin file and load it.
4. Select **Edit > Edit Menu Layout**.
5. From the **Command** list on the left-hand side of the Edit Menu Layout dialog, unroll the **Plug-ins** rollup.
6. Locate the three FBX utilities—**FBX Import**, **FBX Merge** and **FBX Export**—and drag each to an appropriate location in the **Menus** section of the dialog. Common choices are **File > Import** and **File > Export**. You may have to unroll menu entries to locate the appropriate submenu.
7. Click **Done**. The FBX utilities will now be available from the menu locations that you have selected.

Step 2: Preparing your Lightwave character

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

You need to ensure the following:

- Ensure that your Lightwave 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.
- The character should be facing along the **positive Z axis** if you are using the Y axis as the vertical direction. Alternatively, the character should be facing along the **negative Y axis** if you are using the Z axis as the vertical direction.
- Ideally, you should set the Grid Units to **Meters**, and the System Units to **Centimeters**. However, if this is not possible, you can still make scale adjustments during import and export.

Step 3: Terminating LightWave bones

LightWave uses a **bone-based** skeletal definition. Many animation systems and file formats—including *endorphin* and FBX—use **joint-based** definitions.

In a joint-based hierarchy, the skeleton is defined as a set of **joints**. Bones only exist as connections between joints, and are drawn implicitly. This means that in order for a bone to exist there needs to be a start joint, as well as an end joint for that bone. However, for bone-based skeletons—such as LightWave characters—there are no end joints.

For example, the **feet** of a hierarchy created in LightWave might end with **toe** bones, but there will be no final toe joints to terminate these bones. The FBX format requires that there be end joints—otherwise toe bones will not be created.

For this reason, at the end of each extremity of your LightWave hierarchy you will need to create an additional child bone. This can be done by selecting the extremity bone and clicking **Add > Child Bone** from the **Setup** tab. These terminating bones to **not** need to be active.

Step 4: Exporting your character from Lightwave

In this step you will export your character as an **FBX** file.

To save your Lightwave character as an FBX file:

1. Launch Lightwave and open a scene containing the character that you want to simulate in *endorphin*.
2. Select **File > Export > Export to FBX**. Keep in mind that you may have chosen a different location for the Export to FBX command menu item.
3. In the **Export** dialog, specify a filename and location for your FBX file.
4. Turn off the **Materials**, **Embed Textures**, **Cameras** and **Lights** settings. Leave all other settings at their default values.
5. Click **OK** to export the character as an FBX file.

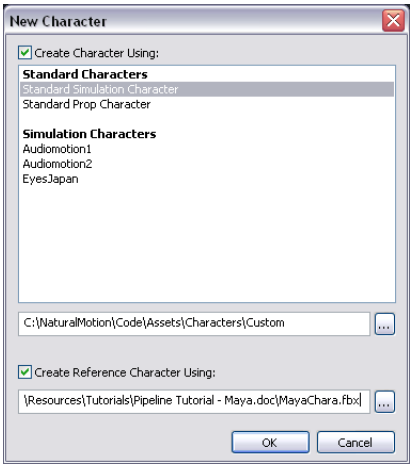
Step 5: Creating corresponding *endorphin* characters

In this step, you will create a **pair** of characters in *endorphin* that represent your Lightwave character. Once you have created this simulation-reference character pair, you will be able to use them to import animation from Lighwave 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 Lightwave character. The other character will be an *endorphin* **reference** character created from the FBX file you saved in Step 4.

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 FBX file that you saved in Step 4. In the **Import Options** dialog, select the root of your FBX character from the joint hierarchy tree, and then click **OK**.



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 you can specify a 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*.

Modifying mass and collision objects

You can also resize and reshape the **mass objects** and **collision objects** of the simulation character, and also create additional collision objects.

A useful technique is to use an imported OBJ mesh file—representing a skin definition—as a guide while modifying the distribution of mass and collision objects.

See the User Guide for more information on using Character Edit Mode to create custom characters.

Step 6: Exporting animation from Lightwave into *endorphin*

This step will be described in a future release of the *endorphin* User Guide. Please contact our technical support team for assistance.

Step 7: Exporting animation from *endorphin* into Lightwave

In this step, you are going to **export** animation that you have created in *endorphin*, and then **import** it into Lightwave, using a 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 **Export To Reference Character** setting.
5. Click the **Browse** button in the Auto Motion Transfer settings. Select the corresponding reference character file you created for exporting data in Step 4.
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 Lightwave:

1. Select **File > Import > Merge With FBX**. Keep in mind that you may have chosen a different location for the Export to FBX command menu item.
2. Select the FBX file that you exported from *endorphin*.
3. Click **OK**. The motion created in *endorphin* will be loaded onto your LightWave character.
4. Click **Play** to see the animation on the your character.

Introducing the Maya pipeline

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

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

The following steps outline the process of transferring data from Maya to *endorphin* and back.

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.
- The character should be facing along the **positive Z axis** if you are using the Y axis as the vertical direction. Alternatively, the character should be facing along the **negative Y axis** if you are using the Z axis as the vertical 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 character to FBX and OBJ files:

1. Launch Maya and open a scene containing your character.
2. Select the Maya character that you want to simulate in *endorphin*.
3. Select **File > Export All**. Choose **FBX** from the File Type list, and specify a file name.
4. Click **Export**. The skeletal structure of your Maya character will be saved into an FBX file.

5. Select **File > Export All**. Choose **OBJ** from the File Type list, and specify a file name. It is generally good working practice to name the FBX and OBJ files using the same name.
6. Click **Export**. The polygonal skin mesh of your Maya character will be saved into an OBJ file.

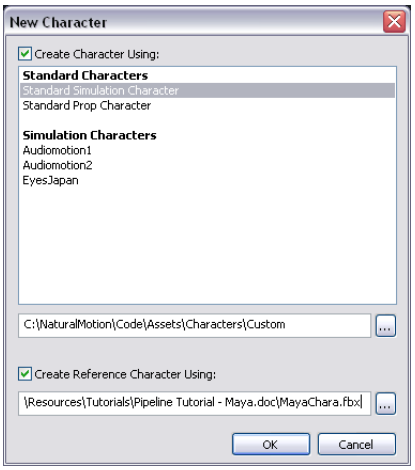
Step 3: Creating corresponding *endorphin* characters

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 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**.



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 you can specify a 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

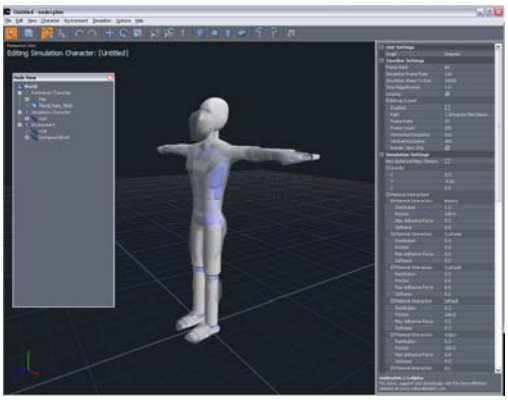
In this step, you will modify the collision objects of the simulation character to fill the volume of the skin mesh saved in the OBJ file in Step 2. You may also add new collision objects as well as modifying existing collision objects. This is often called **filling out** a character.

See the User Guide for more information on using Character Edit Mode to create custom characters.

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 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 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.



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 that contains the animation that you want to export.
2. Select **File > Export All**. Specify a 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 3dsMax with the corresponding character. Click **Open** to display the **Import Options** dialog.
7. In the Import Option dialog, turn on the **Import From Reference Character** setting.
8. Click the **Browse** button in the Auto Motion Transfer settings. Select the corresponding 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 correct root 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 **Export To Reference Character** setting.
5. Click the **Browse** button in the Auto Motion Transfer settings. Select the corresponding 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 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.

Introducing the MotionBuilder pipeline

To reset MotionBuilder degrees of freedom (DOFs) to zero, you need to **Characterize** the character, then apply a **Control Rig**, and then finally **zero** the DOFs with the rig on.

To reset MotionBuilder DOFs:

1. Drag a character from the **Asset Browser** onto your skeleton.
2. Select the newly-created character in the **Navigator** window.
3. In the **Character Definition** window associate all the joints to the rig that you are going to zero out.
4. Once the definition has been prepared, **Characterize** the skeleton and create a **Control Rig**.
5. Switch to the **Character Settings** tab and click **Plot Character**.
6. With the skeleton now controlled by the rig, we can zero the DOFs. In the navigator window, right-click on the root of your character and select **Select Branch**. This will select all the joints.
7. In the **Properties** window, under **Degrees of Freedom > Rotation**, zero all the values for **Pre Rotation** and **Post Rotation**.
8. Once you have done this click on **Plot Character** in the Character Settings tab to plot it back to the skeleton.
9. You can now delete the control rig if required.

Introducing the Poser pipeline

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

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

The following steps outline the process of transferring data from Poser to *endorphin* and back.

Step 1: Exporting your character from Poser

In this step you will export your character as a **BVH** file.

To save your Poser character to a BVH file:

1. Launch Poser and open a scene containing your character.
2. Select your character.
3. Select **File > Export > BVH Motion**.
4. If you are using **Poser 6** you will need to run a script in Poser to remove the **Centre Of Mass** object from the file. This script can be found at **www.curiouslabs.com/article/articleview/1393/1/595/**.

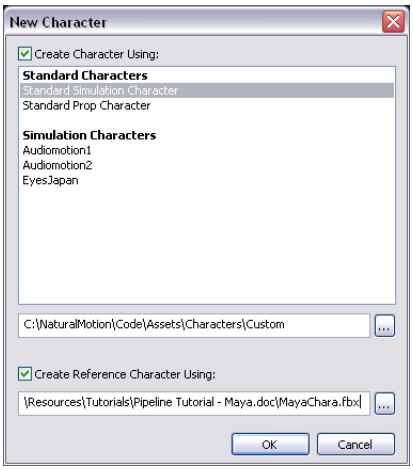
Step 2: Creating corresponding *endorphin* characters

In this step, you will create a **pair** of characters in *endorphin* that represent your Poser character. Once you have created this simulation-reference character pair, you will be able to use them to import animation from Poser 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 Poser 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 BVH file that you saved in Step 1. In the **Import Options** dialog, select the root of your BVH character from the joint hierarchy tree, and then click **OK**.



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 you can specify a 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*.

Modifying mass and collision objects

You can also resize and reshape the **mass objects** and **collision objects** of the simulation character, and also create additional collision objects.

A useful technique is to use an imported OBJ mesh file—representing a skin definition—as a guide while modifying the distribution of mass and collision objects.

See the User Guide for more information on using Character Edit Mode to create custom characters.

Step 3: Exporting animation from Poser into *endorphin*

In this step, you are going to **import** animation that you have created in Poser into *endorphin*, using a BVH file to transfer the data.

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 2. 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 BVH file that you created using Poser with the corresponding character. Click **Open** to display the **Import Options** dialog.
7. In the Import Option dialog, turn on the **Import From Reference Character** setting.
8. Click the **Browse** button in the Auto Motion Transfer settings. Select the corresponding reference character file you created for importing data in Step.
9. In the **General** settings, turn on the **Include parent transforms** setting.
10. In the node hierarchy tree, select the **Hip** node as the root 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 4: Exporting animation from *endorphin* into Poser

In this step, you are going to **export** animation that you have created in *endorphin*, and then **import** it into Poser, using a BVH 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 BVH Files. You can enter a new name, or choose an existing BVH file to replace it. Click **Save** to display the **Export Options** dialog.
4. In the **Export Options** dialog, turn on the **Export To Reference Character** setting.
5. Click the **Browse** button in the Auto Motion Transfer settings. Select the corresponding reference character file you created for importing data in Step 2.
6. Click **OK** to export the animation to the BVH 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 Poser:

1. In Poser, open the scene that you want to import animation into.
2. Select **File > Import > BVH Motion** and browse for the BVH file that you saved from *endorphin*.

Chapter 22

Frequently Asked Questions

Introduction

If you can't find what you are looking for in the endorphin *Reference* Manual, you may find the answer to your question in the following set of **frequently asked questions**. This list will be growing in future releases of this User Guide.

Frequently asked questions:

- Can I use *endorphin* with other animation systems?
- Can I create *endorphin* characters in other animation systems?
- Can I use *endorphin* with characters of different shapes and sizes?
- Can I use *endorphin* with my own character rigs?
- Can I use *endorphin* with multi-legged characters like horses?
- Can I animate *endorphin* characters using keyframing?
- How do I pan, rotate and zoom in the viewport?
- How do I change the shape of mass and collision objects?
- How do I create forces at an offset from the centre of a mass object?
- How do I export the motion of objects in the environment?

Learning Edition:

- How do I use the Learning Edition product key?
- What if I my Learning Edition registration email has not arrived?

Can I use *endorphin* with other animation systems?

Yes. *endorphin* can import and export a range of file formats, including **FBX**, **XSI** and **BVH** files.

This means you can **import** animation into *endorphin* that you have generated in systems such as Maya, 3dsMax, SOFTIMAGE|XSI, MotionBuilder, LightWave and Poser. Similarly, once you have generated animation in endorphin, you can **export** it for use in Maya, 3dsMax, SOFTIMAGE|XSI, MotionBuilder, LightWave, Poser and other animation systems.

endorphin can also import and export the **Vicon** motion capture file format.

Can I create *endorphin* characters in other animation systems?

You can create *endorphin* **prop characters**—such as vehicles and weapons—in **Maya** and **3dsMax**. *endorphin* ships with various third-party scripts that allow you to create prop characters in these animation systems and save them as endorphin **.nmc** characters. The third-party scripts are located in **Resources\Third Party\Scripts**.

- The MEL script **Maya\endorphin_EnvironmentPropExporter.mel** allows you to create *endorphin* prop characters in Maya.
- The 3dsMax script **3dsMax\endorphin_EnvironmentPropExporter.ms** allows you to create *endorphin* prop characters in 3dsMax.

To create *endorphin* **simulation characters**, you must use the *endorphin* Character Edit Mode. In future releases of endorphin, we are planning on extending the third party scripts to allow you to create endorphin simulation characters in other animation systems.

Can I use *endorphin* with characters of different shapes and sizes?

Yes. The *endorphin* standard simulation character is an average-sized adult human male. However, you can use the *endorphin* **Character Edit Mode** to change the shape and size of this character. For example, you can make the character taller or shorter, lighter or heavier, male or female. You can adjust the length of every bone, and the range of motion of every joint. You can also add weapons, clothing, helmets and other objects to the character.

endorphin ships various example characters in the folder **Resources\Characters\Custom**. In addition, many of the sample scenes in the folder **Resources\Scenes\Sample Scenes** include characters of different shapes and sizes, such as soldiers, footballers and gymnasts.

Can I use *endorphin* with my own character rigs?

Indirectly. The characters used in an *endorphin* scene are always derived from the *endorphin* **standard simulation character**. This means that all *endorphin* characters—no matter how large or small, light or heavy, dwarf or soldier—are all based on the same internal joint hierarchy.

When you create new *endorphin* simulation characters in **Character Edit Mode**, you can add your own character as the **reference character** of the simulation character. *endorphin* uses a technique called **dynamic motion transfer** to map motion from your reference character to its corresponding endorphin simulation character, and vice versa.

Can I use *endorphin* with multi-legged characters like horses?

Yes and no. If you have modelled a multi-legged character—such as a horse—in another system like **3dsMax** or **Maya**, you can create a corresponding **prop character** in *endorphin* based on this character. You can then use this character like any other prop character in an *endorphin* scene.

For example, if you have modelled a **horse** in another animation system, you can use Character Edit Mode to create a corresponding *endorphin* horse prop character. By carefully modeling the distribution of mass and collision objects—and also carefully setting the correct range of motion for each joint—you can create very realistic equine motion in *endorphin*.

As for simulation characters, you can apply forces, constraints, active poses and sever events to prop characters. You can also import animation for prop characters, and also use simulation and transition events.

Keep in mind that you **cannot** use behaviours on prop characters. The *endorphin* Behaviour Library has been developed for bipedal (two-legged) characters derived from the *endorphin* standard simulation character.

Can I animate *endorphin* characters using keyframing?

Yes. *endorphin* generates animation using physical simulations. However, within each scene, you can choose which characters are **simulated**, and which characters are driven by **keyframes**. For example, you may have an *endorphin* simulation character colliding with a prop character such as vehicle. The simulation character may be simulated, whereas the vehicle's motion may be entirely driven by keyframes.

You can also change the **simulation mode** for each character at different times, allowing characters to be physically simulated over some frame ranges, and keyframed over other frame ranges. For example, you may use an imported keyframed data such as a walk cycle to drive a character for a number of frames, You could then **simulate** the character once a particular frame is reached in order to dynamically generate complex dynamic motion. At a later frame, you could then **transition** back into another keyframed mode.

How do I pan, rotate and zoom in the viewport?

endorphin includes a number of different ways to manipulate the viewport camera.

If you have a **three-button mouse** that includes a mousewheel, use the **middle** mouse button to **rotate**; the **middle** and **right** mouse buttons together to **pan**; and the mousewheel to **zoom**. This approach is commonly-used in many 3D editing environments, and is particularly useful because it does not require using the keyboard.

endorphin also supports the **Maya** camera approach. If you have a **three-button mouse**, hold down the **Alt** key and use the **left** mouse button to **rotate**; the **middle** mouse button to **pan**; and the **right** mouse button to **zoom**. Alternatively, if you have a **two-button mouse**, you can hold down the **Ctrl** and **Alt** keys together and use the **left** mouse button to **pan**.

You can also use the **Plus (+)** and **Minus (–)** keys to **zoom in** and **zoom out**.

How do I change the shape of mass and collision objects?

Mass and collision objects are available in four shapes—**cylinders**, **spheres**, **boxes** and **sphyls**. Sphyls are capsules with cylindrical bodies and spherical end-caps. Boxes are also called cubiods.

To change the shape of a mass object, select the object in the viewport or in the Node View. In the Property Editor, locate the **Type** property, which is located in the **Geometry** property set. You can use the Type combobox to change the shape of the object. You can change the shape of multiple selected objects at the same time.

Keep in mind that you can only change the shape of mass and collision objects belonging to the Environment, unless you are in **Character Edit Mode**. In Character Edit Mode, you can change the shape or mass and collision objects belonging to the character.

How do I create forces at an offset from the centre of a mass object?

When you apply forces to mass objects, the force is always applied so that it passes through the **centre of mass** of the mass object. You cannot directly apply a force at an offset to the centre of mass. There are times where this would be useful—applying a force away from the centre of mass allows you to generate a **rotational** force as well as the usual translational force.

A useful technique to achieve this effect is to create a small **helper mass object**, and place this helper mass object at the desired force position. By applying a force to the helper mass object, and **constraining** the helper mass object to the original mass object, you can effectively generate a rotational force. You can experiment with different helper object sizes and densities to increase or decrease the relative strength of the rotational force generated.

How do I export the motion of objects in the environment?

In many cases, you would like to be able to export the motion of *endorphin* environment objects, particularly when there is complex interaction between simulation characters and environment objects.

If you are using the FBX file format, you can export the motion of the Environment just as for any other character—select the Environment character, and then select **File > Export**.

If you are exporting the motion of any other character using the FBX file format, you can turn on the **Export Environment** setting in the **Export Options** dialog to include the motion of the environment objects in the same FBX file.

How do I use the Learning Edition product key?

You can access your *endorphin* **Learning Edition product key** at any time using your **User Community Forum profile**. You will need to use this key the first time you run *endorphin* Learning Edition after installation.

To obtain and use your Learning Edition product key:

- Follow this link to access your community forum profile, and scroll down to locate your product key. You can copy and paste this key directly into the Product Key field of the **License Key** dialog that is displayed the first time you run *endorphin* Learning Edition after installation.
- Be careful to ensure that when entering your email address into the **License Key** dialog that you enter **exactly** the same email address that you used to register your profile with the *endorphin* User Community Forum. Your product key will only work in conjunction with that email address.
- There is no limit on how many times you can use your product key.

What if I my Learning Edition registration email has not arrived?

Due to the vagaries of email systems beyond our control, it can take up to 24 hours for your email message to arrive. If your registration email still hasn't arrived, then there may have been a problem with your registration.

Troubleshooting lost registration email:

- Follow this link to check the status of your registration. If you are asked to **login**, please login with the username and password you gave when you initially registered.

- Follow the **Edit Email & Password** link. Please check that the email address you have given is valid. If it isn't, then you can alter it here and a new activation email will automatically be sent out to you.
- Check that the email account you have is still active and isn't full. This may seem obvious, but we have had many failed message deliveries as a result of full or disabled email accounts.
- Learning Edition activation emails are occasionally mistakenly identified as **spam** by some over-sensitive message filters. If you have such a filter installed, it may be worth checking your **suspected junk mail** folders in case it has ended up in there.
- If you still do not receive your registration email message, please contact us. Make sure that you provide the same email address you gave when you registered, and we'll see if we can help.

Chapter 23

Troubleshooting

Introduction

If you are having problems running *endorphin*, please check the following guide. It offers advice for many common issues that occur. Most issues are related to configurations of hardware, software, operating system and system drivers.

You should also check the NaturalMotion site for software updates, and check the *endorphin* User Community Forum for updates and advice. If you are still unable to correct the problem, please contact our technical support team directly via email or telephone.

Troubleshooting guide:

- Checking your system specification
- Checking your graphics driver version
- Updating your graphics drivers
- Avoiding anti-aliasing problems
- Avoiding multithreading problems
- Known issues

Checking your system specification

endorphin is a complex, multithreaded, graphics-intensive application. It is important to have the correct hardware, operating system, system driver and software configuration before attempting to run *endorphin*. It is particularly important to ensure that you are using a supported graphics card and graphics driver.

The following symptoms may indicate problems with your hardware, software or driver configuration:

- *endorphin* periodically shuts down with the error message ***endorphin* has performed an illegal operation and will be shut down**;
- *endorphin* becomes unresponsive, particularly with mouse input or viewport graphics refreshing;

- *endorphin* displays graphical artifacts or corruptions in the viewports.

If you notice any of these problems, you should check that your system conforms to the supported settings.

You should also check the NaturalMotion site for software updates, and check the *endorphin* User Community Forum for updates and advice. If you are still unable to correct the problem, please contact our technical support team directly via email or telephone.

To check your system specification:

To confirm that your system matches the minimum specification, it is important to have an accurate, comprehensive listing of your system specifications. You should send your system specification to us when contacting technical support or when posting software bug reports. Most recent versions of Microsoft Windows include the **msinfo32.exe** utility. This tool generates a simple text file containing your system information.

1. Click **Start > Run** on the Windows taskbar. This displays the **Run** dialog.
2. Type **msinfo32.exe** in the **Open** textbox, and click **OK**.
3. The **System Information** dialog is displayed. You can browse through this dialog to review your system settings.
4. To save your system settings as a text file, select **File> Export** (if you are running **Windows XP**) or **Action > Save As Text File** (if you are running **Windows 2000**). Browse for a destination folder and filename, and click **OK**.

Checking your graphics driver version

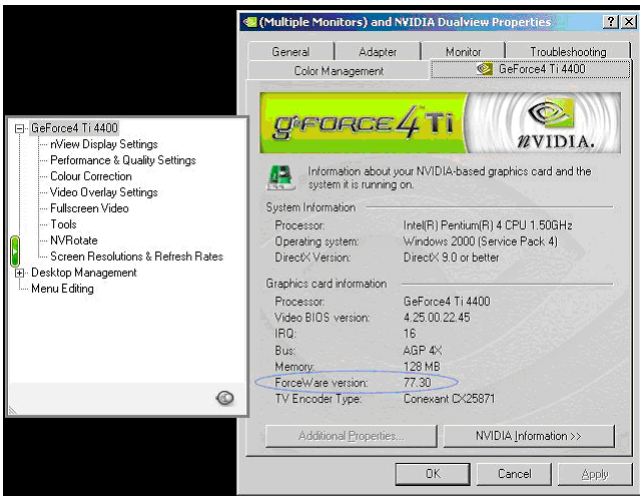
A common cause of *endorphin* instability problems is out-of-date or unsupported graphics cards or graphics device drivers. Ensuring your graphics card and drivers and valid and up-to-date is probably the single-most important way of improving your *endorphin* system stability.

To check your graphics driver version using the Control Panel:

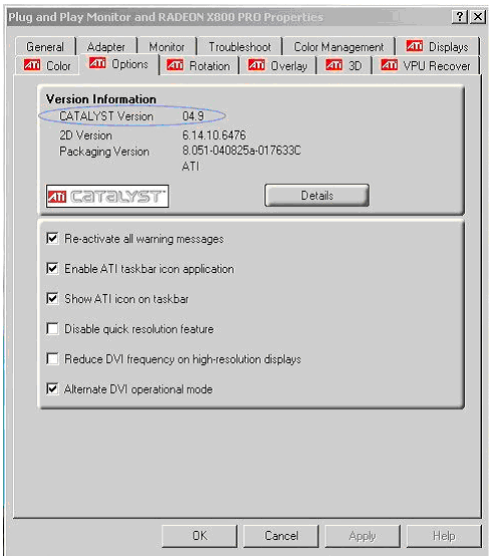
The method of checking for your graphics driver version will vary depending on your system. If you are using an nVidia or ATI graphics card, the following approach should work:

1. Click **Start > Control Panel** on the Windows taskbar. This displays the **Control Panel** dialog.
2. Double-click on the **Display** item. This displays the **Display Properties** dialog.
3. Browse to the **Settings** tab.
4. Click the **Advanced** button.

5. If you are using an **nVidia** graphics card, one of the tabs should display the nVidia logo. **Click** on this tab. You should see a dialog similar to this one displayed below. The nVidia driver version is listed as the **ForceWare version**. For older nVidia graphics cards, the version may be listed as the **Detonator version**.



If you are using an **ATI** graphics card, you should see many additional tabs should displaying the **ATI** logo. **Click** on the tab labeled **Options**. You should see a dialog similar to this one displayed below. The ATI driver version is listed as the **CATALYST version**.



To check your graphics driver version using the System Information dialog:

You can also use the System Information dialog to identify your graphics driver version.

1. Click **Start > Run** on the Windows taskbar. This displays the **Run** dialog.
2. Type **msinfo32.exe** in the **Open** textbox, and click **OK**.
3. The **System Information** dialog is displayed. You can browse through this dialog to review your system settings.
4. Select **Components > Display** in the System Summary panel.
5. Browse to locate the **Driver Version** item.
6. The **Value** property of the Driver Version indicates your graphics driver version.
    - If you are using an **nVidia** graphics card, the last four digits of the reported driver version correspond to the publicly-defined driver version. For example, the reported driver version **6.14.10.7730** is equivalent to the publicly-defined driver version **77.30**.
    - If you are using an ATI graphics card, there is no direct obvious correlation between the reported driver version and the public-defined driver version. For example, the reported driver version 6.14.10.6553 is equivalent to the Catalyst 5.7 driver. Consult the Tech Power-Up site for more details on ATI driver versions.

Updating your graphics drivers

To install new versions of a display driver, download the relevant installer from ATI or nVidia and run it.

Most notebook PC manufacturers have similar systems for their drivers. It is generally recommended that you uninstall the current drivers prior to installing any new ones. You can do this through **Add/Remove Programs** in the Windows **Control Panel**.

It can be difficult for **notebook PC** users to obtain up-to-date graphics drivers, as many manufacturers will stop releasing updates for systems that have reached a certain maturity—even where there are known issues with the latest graphics drivers. Unfortunately we are unable to assist in such cases. There are some methods for adapting some desktop PC graphics drivers to work with certain models of notebook PC video hardware. However, since these methods are complex, potentially unreliable and definitely unsupported by the hardware manufacturers, we can't publish them ourselves—or even officially recommend their use. Web searches can help here if you would to locate these yourself.

Avoiding anti-aliasing problems

Most modern graphics systems support **anti-aliasing**. This is a technique for visually smoothing out the display to remove pixelation effects. Typically, graphics systems will provide different levels of anti-aliasing, with an option for the user to override the default setting.

Some users have reported *endorphin* instability when using the anti-aliasing feature of some ATI cards, particularly when using the Pose Move and Pose Rotate tools, and when creating force events. If you are experiencing endorphin instability whilst using an ATI graphics card, we recommend manually turning **off** the anti-aliasing setting. You may notice some slight graphics degradation.

Avoiding multithreading problems

endorphin is a multithreaded application. This means that if you have a multiple-processor or dual-core CPU, *endorphin* is able to run more efficiently.

Some users have reported instabilities and other problems when running *endorphin* in multithreaded mode, particularly when using ATI graphics cards. If you find that *endorphin* viewports freeze sporadically during simulation or playback, try running *endorphin* in singlethreaded mode.

To change the *endorphin* threading mode:

1. Select **Options > Display**. This displays the **Display Options** dialog.
2. Browse to select the **View** tab.
3. Select the **Single-threaded** or **Multithreaded** settings to run *endorphin* in the specified mode. You may notice some performance degradation using the single-threaded option. Only use the single-threaded setting if definitely experiencing problems in multithreaded mode.

Known issues

We welcome feedback from all users. If you have any technical problems or bug reports, please contact our **technical support team** to report the issue. You can also post comments, feedback and bug reports on the ***endorphin* User Community Forum**. We strive to correct issues as soon as they appear.

Ensure that you always use the latest release of *endorphin*. If you are using *endorphin* Learning Edition, an **Upgrade** dialog will automatically be displayed when new releases of endorphin are made available (provided that you have an internet connection).

Some of the issues we are aware of in *endorphin* 2.6 include:

- **FBX import of animated joint lengths.** When importing an FBX file containing animated joint lengths the lengths are currently drawn incorrectly.
- **Node View multiple selection.** When selecting multiple objects in the node view, only the icon for the last selected object is initially displayed in the highlight colour.
- **Collision objects as behaviour targets.** In Collision Only mode, collision objects must be used as the target objects in the two Reach behaviours, and the Tackle behaviour. Using the new Collision With Momentum mode, mass objects can now be used as targets.
- **Behaviour copy-and-paste.** Copying and pasting behaviours on the timeline is not fully supported. We recommend that you create behaviours explicitly rather than by copying other existing behaviours.
- **Overlapping behaviours.** Overlapping behaviours and active poses that don't use exclusive selection sets can result in the behaviours not activating.
- **Selecting joint limits.** On some older graphics cards it may not be possible to select joint limits by clicking on their representation in the viewport. Selecting joint limits in the Node View is unaffected by this problem.
- **Multiple selection.** When using the move or rotate tool, multiple select can be unreliable. Use the explicit select tool to create selection and then return to previous tool.
- **Saving character rig groups.** It is not possible to save rig groups to a pre-*endorphin* 2.5 character. Upgrading the character will make this functionality available.
- **Character upgrading.** Upgrading characters that have pre-*endorphin* 2.0 behaviours on their timeline can result in the behaviour events having their types reset.
- **Mass object icons in the Node View.** Occasionally, mass object icons in the Node View are drawn as collision object icons. This only affects mass objects contained in some older reference characters. It does not affect mass objects in simulation or prop characters, or in newly-created reference characters.

Resolved issues

- The **Edit Character In Scene** command now works correctly. Previously, after editing characters directly in a scene using this command, you may have found that they were unable to simulate correctly during subsequent simulations. This issue was introduced in *endorphin* 2.6 and has been resolved in *endorphin* 2.6.1.
- The **Use Default Ground** setting now works correctly. Previously, turning off this setting was having no effect on the ground collision object. This issue was introduced in *endorphin* 2.6 and has been resolved in *endorphin* 2.6.1.

End-user license agreement

NATURALMOTION SOFTWARE LICENCE AGREEMENT

UNITED KINGDOM

IMPORTANT, PLEASE READ THIS FIRST. THIS IS A LICENSE AGREEMENT.

NATURALMOTION IS WILLING TO LICENSE THE ACCOMPANYING SOFTWARE TO YOU ONLY UPON THE CONDITION THAT YOU ACCEPT ALL OF THE TERMS CONTAINED IN THIS LICENCE AGREEMENT AND ANY SUPPLEMENTARY OR UNIQUE LICENCE TERMS INCLUDED HEREWITH (**AGREEMENT**).

READ THE TERMS AND CONDITIONS OF THIS LICENCE AGREEMENT CAREFULLY BEFORE SELECTING THE **ACCEPT** BUTTON AT THE BOTTOM OF THE PAGE. PLEASE USE THE SCROLL BAR ON THE RIGHT TO READ THE REST OF THIS LICENCE AGREEMENT. BY SELECTING THE **ACCEPT** BUTTON, YOU ARE CONSENTING TO BE BOUND BY ALL THE TERMS OF THE AGREEMENT AND THE SOFTWARE WILL BE INSTALLED.

IF YOU ARE NOT WILLING TO BE BOUND BY THIS AGREEMENT AND YOU DO NOT AGREE TO ALL OF ITS TERMS AND CONDITIONS, SELECT **DO NOT ACCEPT**, WHICH WILL CANCEL THE LOADING OF THE SOFTWARE. YOUR USE OF THE SOFTWARE ALSO INDICATES YOUR ASSENT TO BE BOUND BY THE TERMS AND CONDITIONS OF THIS AGREEMENT.

COPYING OR USE OF THIS SOFTWARE OR ITS DOCUMENTATION EXCEPT AS PERMITTED BY THIS AGREEMENT IS UNAUTHORISED AND IS COPYRIGHT INFRINGEMENT UNDER THE LAWS OF YOUR COUNTRY. IF YOU COPY OR USE THIS SOFTWARE OR ITS DOCUMENTATION WITHOUT PERMISSION OF NATURALMOTION, YOU ARE VIOLATING THE LAW. YOU MAY BE LIABLE TO NATURALMOTION FOR DAMAGES, AND YOU MAY BE SUBJECT TO CRIMINAL PENALTIES.

1 GRANT OF LICENSE

NaturalMotion Ltd (**NaturalMotion**) grants you a nonexclusive, nontransferable license to use this program (the **Software**) and its manual and other accompanying printed material and **online** or electronic documentation (**Documentation**) with equipment owned by you or under your control, according to the terms and conditions of this Agreement. This Agreement permits a single user to install and use the Software on only one computer at one location at any one time.

Backup Copy: Regardless of which version of the Software you have acquired, you may make one archival (backup) copy of the Software. Such archival copy may not be installed on another computer, unless such computer is a partitioned drive of a server to which only the authorised user has access. In any event, the archival copy may not be used or installed as long as another copy of the Software is installed on any computer. If the Documentation is in printed form, it may not be copied. If the Documentation is in electronic form, it may not be duplicated electronically, however, you may print out one (1) copy, which may not be copied.

Additional Installation: You may make a second copy of the Software on the hard disk of a second computer owned by you or under your control provided that (1) the original and second copies are used only by the same person; (2) the second copy is installed and used only on a notebook computer, home computer, or other non-server computer away from your usual work location for the purpose of enabling you to perform work while away from your usual work location; (3) only one of the Software copies is in use at any one time; and (4) the second copy of the Software is used exclusively with the copy protection device (if any) supplied with the Software.

Upgrades: If this Software is labelled as an upgrade (**New Version**) to software previously licensed to you (**Previous Version**), you must destroy all copies of the Previous Version, including any copies resident on your hard disk drive, and return the hardware lock, if any, which accompanied the software previously licensed to you (unless NaturalMotion explicitly notifies you that the hardware lock is to be used with the upgrade), within sixty (60) days of acquiring the New Version. NaturalMotion reserves the right to require you to show satisfactory proof that the Previous Version has been destroyed. If the hardware lock is not returned within the stipulated period, NaturalMotion reserves the right, without limitation, to charge you, and you shall pay the difference in price between the New Version price and the suggested retail price of the Software. Software patches, if any, provided to you by NaturalMotion or an authorised third-party in connection with the Software licensed to you hereunder shall be subject to the terms and conditions of this Agreement unless otherwise specified at the time of delivery. Notwithstanding the foregoing, you may retain and need not destroy the Previous Version and may use the Previous Version solely if necessary for the purposes of (1) installing the New Version hereby licensed and (2) for archival (backup) purposes in order to reinstall the New Version hereby licensed if the initial installation fails. Under no circumstances may you operate the Previous Version.

Authorisation Code: If this Software requires an authorisation code, you must register your purchase of this Software with NaturalMotion before an authorisation code shall be issued to you, and NaturalMotion shall maintain your registration details.

Licence Term: Subject to the terms and conditions of this Agreement, the licence to use the Software is perpetual, unless the Software is designated as a fixed-term licence, a limited duration licence or a rental licence, and in such case the term of the licence shall be the term for which you have paid.

2 RESTRICTIONS

You may not:

1. copy or use the Software or Documentation except as permitted by this Agreement.
2. reverse engineer, decompile, or disassemble the Software except to the extent permitted by law where this is indispensable to obtain the information necessary to achieve interoperability of an independently created program with the Software or with another program and such information is not readily available from NaturalMotion or elsewhere.

3. distribute, rent, loan, lease, sell, sublicense, or otherwise transfer all or part of the Software, Documentation or any rights granted hereunder to any other person without the prior written consent of NaturalMotion.
4. install or use the Software on the Internet or over a wide area network, including, without limitation, use in connection with a Web hosting or similar service.
5. remove, alter, or obscure any proprietary notices, labels, or marks from the Software or Documentation.
6. modify, translate, adapt, arrange, or create derivative works based on the Software or Documentation for any purpose.
7. utilise any equipment, device, software, or other means designed to circumvent or remove any form of copy protection used by NaturalMotion in connection with the Software, or use the Software together with any hardware lock, authorisation code, serial number, or other copy protection device not supplied by NaturalMotion directly.
8. export the Software or Documentation in violation of US or other applicable export control laws.
9. use the Software or Documentation outside of the country of purchase, unless the Software and Documentation were purchased in the European Union (EU) or the European Economic Area (EEA), in which case use throughout the EU and the EEA is permitted.

3 COPYRIGHT

Title and copyrights to the Software, Documentation and accompanying materials, if any, and any copies made by you remain with NaturalMotion and its licensors. The structure, organisation, and code of the Software are valuable trade secrets of NaturalMotion and its licensors. Unauthorised copying of the Software or Documentation, or failure to comply with the above restrictions, will result in automatic termination of this Agreement. This Agreement does not grant you any intellectual property rights.

4 GENERAL LIMITED WARRANTY

NaturalMotion warrants that the Software will provide the facilities and functions generally described in the Documentation and that the media on which the Software is furnished, if any, the Documentation accompanying the Software, and any hardware lock or other copy protection device accompanying the Software will be free from defects in materials and workmanship under normal use. EXCEPT FOR THE ABOVE EXPRESS LIMITED WARRANTIES, NATURALMOTION MAKES AND YOU RECEIVE NO WARRANTIES, EXPRESS, IMPLIED, STATUTORY, OR IN ANY COMMUNICATION WITH YOU, AND NATURALMOTION SPECIFICALLY DISCLAIMS ANY OTHER WARRANTY INCLUDING THE IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. NATURALMOTION DOES NOT WARRANT THAT THE OPERATION OF THE SOFTWARE WILL BE UNINTERRUPTED OR ERROR FREE. The above exclusions may not apply to you as some jurisdictions do not

allow the exclusion of implied warranties. In addition to the above warranty rights, you may also have other rights, which vary from jurisdiction to jurisdiction.

NaturalMotion's entire liability and your exclusive remedy under the warranties made in this Agreement will be, at NaturalMotion's option, to attempt to correct or work around errors, to replace the defective media, if any, Documentation or copy protection devices; or to refund the purchase price and terminate this licence. This remedy is subject to the return of the defective media, documentation, or copy protection device with a copy of your receipt to NaturalMotion within ninety (90) days from the date of its delivery to you. Following expiration of this ninety (90)-day period, NaturalMotion will replace any defective or damaged copy protection device in return for payment of an amount that covers the cost of a replacement device plus a fee for handling and shipment.

5 DISCLAIMER

3D ANIMATION SOFTWARE AND OTHER TECHNICAL SOFTWARE ARE TOOLS INTENDED TO BE USED BY TRAINED PROFESSIONALS ONLY. THEY ARE NOT SUBSTITUTES FOR YOUR PROFESSIONAL JUDGEMENT. DUE TO THE LARGE VARIETY OF POTENTIAL APPLICATIONS FOR THE SOFTWARE, THE SOFTWARE HAS NOT BEEN TESTED IN ALL SITUATIONS UNDER WHICH IT MAY BE USED. NATURALMOTION SHALL NOT BE LIABLE IN ANY MANNER WHATSOEVER FOR THE RESULTS OBTAINED THROUGH THE USE OF THE SOFTWARE. PERSONS USING THE SOFTWARE ARE RESPONSIBLE FOR THE SUPERVISION, MANAGEMENT AND CONTROL OF THE SOFTWARE. THIS RESPONSIBILITY INCLUDES, BUT IS NOT LIMITED TO, THE DETERMINATION OF APPROPRIATE USES FOR THE SOFTWARE AND THE SELECTION OF THE SOFTWARE AND OTHER PROGRAMS TO ACHIEVE INTENDED RESULTS. PERSONS USING THE SOFTWARE ARE ALSO RESPONSIBLE FOR ESTABLISHING THE ADEQUACY OF INDEPENDENT PROCEDURES FOR TESTING THE RELIABILITY AND ACCURACY OF ANY PROGRAM OUTPUT.

6 LIMITATION OF LIABILITY

IN NO EVENT WILL NATURALMOTION BE LIABLE FOR ANY LOSS OR DAMAGES OF ANY KIND, INCLUDING LOSS OF DATA, LOST PROFITS, COST OF COVER, OR OTHER SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR INDIRECT DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE SOFTWARE OR DOCUMENTATION, HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY. THIS LIMITATION WILL APPLY EVEN IF NATURALMOTION HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE. YOU ACKNOWLEDGE THAT THE LICENCE FEE REFLECTS THIS ALLOCATION OF RISK.

NATURALMOTION SHALL HAVE NO RESPONSIBILITY OR LIABILITY WHATSOEVER ARISING FROM LOSS OR THEFT OF THE SOFTWARE OR OF ANY COPY PROTECTION DEVICE WITH WHICH THE SOFTWARE IS SUPPLIED. SPECIFICALLY, NATURALMOTION SHALL NOT BE OBLIGATED TO REPLACE ANY LOST OR STOLEN SOFTWARE OR COPY PROTECTION DEVICE. YOU ARE SOLELY RESPONSIBLE FOR SAFEGUARDING THE SOFTWARE AND ANY COPY PROTECTION DEVICE FROM LOSS OR THEFT AND PROTECTING YOUR INVESTMENT THROUGH INSURANCE OR OTHERWISE.

THE ABOVE LIMITATIONS MAY NOT APPLY TO YOU BECAUSE SOME JURISDICTIONS DO NOT ALLOW THE LIMITATION OR EXCLUSION OF LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES.

7 GENERAL

A. This Agreement and the licence granted hereby shall terminate without further notice or action by NaturalMotion if you, the licensee, become bankrupt, make an arrangement with your creditors or go into liquidation.

B. This Licence Agreement shall not be governed by the UN Convention on Contracts for the Sale of Goods. This Licence Agreement shall be governed by the laws of England without reference to conflict-of-laws principles. This Licence Agreement is the entire agreement between us and supersedes any other communications or advertising with respect to the Software and Documentation.

C. If any provision of this Agreement is found to be invalid or otherwise unenforceable, the further conditions of this Agreement will remain fully effective and the parties will be bound by obligations which approximate, as closely as possible, the effect of the provision found invalid or unenforceable, without being themselves invalid or unenforceable.

26 April 2004

NaturalMotion Ltd.

Beaver House,

Hythe Bridge Street

Oxford, Oxfordshire OX1 2ET

United Kingdom

Index

3

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

Z