

Cult3D 5.3

User Guide Supplement

Contents

CULT3D 5.3 INTRODUCTION.....	4
WHAT'S NEW IN BRIEF	4
USING AND LEARNING CULT3D	6
<i>Cult3D 5.3 Software Package.....</i>	<i>6</i>
<i>Documentation.....</i>	<i>7</i>
<i>Workflow and Useful Resources.....</i>	<i>7</i>
<i>System Requirements.....</i>	<i>9</i>
<i>Software and Hardware Rendering.....</i>	<i>9</i>
NEW FEATURES DOCUMENTATION	11
CULT3D RENDER.....	11
<i>Antialiasing, new Progressive Mode.....</i>	<i>11</i>
<i>Bump Mapping.....</i>	<i>12</i>
CULT3D EXPORTER	13
<i>MaxScript Export to Cult3D.....</i>	<i>13</i>
CULT3D DESIGNER	15
<i>Cult3D Stereo for Objects.....</i>	<i>15</i>
<i>Walkthrough and Collision Detection.....</i>	<i>22</i>
<i>Extended Arcball Action.....</i>	<i>29</i>
<i>Tools menu.....</i>	<i>32</i>
<i>The Workbench Window.....</i>	<i>33</i>
<i>The Wizard.....</i>	<i>35</i>
<i>Multiple Selection in the Scene Graph.....</i>	<i>36</i>
<i>Helpers in the Event Map.....</i>	<i>36</i>
<i>Metrics Tools in the Designer.....</i>	<i>37</i>
<i>Geometries Compression Settings.....</i>	<i>39</i>
<i>Automatic HTML file creation.....</i>	<i>40</i>
UPDATES TO THE CULT3D JAVA API	42
CULT3D.JS FILE AND JAVASCRIPT	45
<i>Cult3D.js File.....</i>	<i>45</i>
<i>Suggested HTML code.....</i>	<i>45</i>
SOFTWARE UPDATES	49
TOGGLE, SWAP ACTIONS.....	49
MANIPULATOR.....	49
SELECTING OBJECTS FROM PREVIEW WINDOW.....	50
5.2 FEATURES - EXTENDED DOCUMENTATION	51
WORLDS	51
ARCBALL FRICTION DURATION	55
HOTSPOTS.....	55
EVENTS WINDOWS.....	56
<i>Events Tool Window.....</i>	<i>56</i>
<i>Edit Event Dialog Box.....</i>	<i>56</i>
<i>Time Line Tool.....</i>	<i>58</i>
CULT3D AND CULT3D JAVA.....	60
<i>Why Java in Cult3D.....</i>	<i>60</i>

<i>How to create Cult3D Java.....</i>	<i>60</i>
<i>How to connect Cult3D and Cult3D Java.....</i>	<i>62</i>
CULT3D EXTERNAL COMMUNICATION.....	69
APPENDIX A: THE COLORCODE 3-D™ TECHNOLOGY INTEGRATED IN CULT3D	70
TRADEMARK ACKNOWLEDGEMENT	71

Cult3D 5.3 Introduction

What's New in Brief

Cult3D 5.3 speeds up presentations development and introduces new features, which increase the visual effect even more.

Cult3D Objects can now be viewed in high quality **3D Stereo**, adding true depth and perspective to Cult3D's visual and interactive realism. By using the **ColorCode 3-D™** technology, Cult3D 5.3 has been enhanced with the following features:

- Exceptional 3D Stereo images can be viewed on any Cult3D supported computer.
- 3D Stereo can be easily applied to any Cult3D object, without sacrificing visual quality, animations, interactivity or any other Cult3D feature.
- No additional hardware or 3D acceleration is required to view 3D Stereo images. No need for expensive or heavy 3D Stereo viewing equipment, just an inexpensive ColorCodeViewer™ with special designed amber and blue filters.

The Arcball behaviour customisation, like limited rotations, is now easily implemented with the new **Extended Arcball**.

Collision Detection and **Walkthrough** actions are appealing new features especially useful in architectural applications of the Cult3D technology.

The new **Progressive Antialiasing** option furthermore enhances the Cult3D visualization realism without compromising the object responsiveness.

Cult3D now supports **Bump Mapping** in hardware.

New tools in the Cult3D Designer like **Multiple Selection**, **Event Map helpers**, **Metrics**, **Resize/Center** tool, **Visibility of Objects** tools, project creation **Wizard** and **Workbench Window** will speed up the Cult3D Designer phase.

Compression customisation has been improved with separate sliders for compressing mesh vertices and UV coordinates.

The Cult3D exporter for 3ds max™ is now scriptable. Cult3D *.c3d files can be created with **MaxScript** without having to interact with the exporter window.

The Designer automatically generates an **HTML file** when saving Cult3D Internet files.

The **Java API** has been extended. Java programmers' control has been empowered with new methods dealing with: mouse, camera, materials, events and tooltips.

Interacting with Cult3D objects in HTML pages, using **JavaScript**, is much easier thanks to the new Cult3D.js file.

Using and Learning Cult3D

Cult3D 5.3 Software Package

What you find in the folder where you installed the Cult3D Designer 5.3:

- *C3DDesigner.exe* - Cult3D Designer 5.3 Application.
- *Classes* folder - contains Cult3D Java classes used in some of the projects in the *Projects* folder.
- *HTML* folder - contains basic Cult3D HTML code.
- *JavaAPI* folder - contains the full documentation for the Cult3D Java development kit.
- *JavaSamples* folder - contains Cult3D Java samples.
- *New 5.3 Samples* folder - contains samples of new 5.3 features.
- *Projects* folder - contains projects used in basic tutorials which can be found in the Cult3D 5.2 documentation, and some self-explanatory projects.
- *Sounds* folder - contains sounds used in basic tutorials, see above.
- *Users Guide* folder - contains Cult3D 5.2 documentation and the Cult3D 5.3 User Guide Supplement.
- *Cult3D JS File* folder - contains the new improved and enhanced Cult3D.js file, which is used for JavaScript communication with Cult3D.
- *Cult3DDevelop.jar* file - is the Cult3D Java development API file. Read the Cult3D Java documentation on how to use it (see *JavaAPI* folder above).
- *ReadMe Java Developers.txt* file - is a text file that Java developers might want to read.

Documentation

This documentation is a Supplement to the Cult3D Designer 5.2 User Guide.

Documentation is available online at the following sites:

<http://www.cult3d.com/>

<http://www.worldof3d.com/>

Documentation is also part of the package downloaded with Cult3D software.

Please always refer to Cult3D website for the most updated information.

Workflow and Useful Resources

With the easy Cult3D Designer authoring tool it is fast and simple to create multi-platform high-quality presentations for improving internal, external, online and offline communication.

There are 3 steps to be followed in order to create Cult3D presentations (you can read more at www.cult3d.com):

1. Export from your modelling tool.

Create a model in 3ds max™, plasma™, Autodesk® VIZ™, Maya® and export it from the host program using the appropriate Cult3D Exporter.

You will obtain a *.c3d file (Cycore proprietary file).

2. Add interactivity in the Cult3D Designer.

Open the *.c3d file in the Cult3D Designer, add interaction and save the internet file.

You will obtain the final Cult3D Object, a *.co file (Cycore proprietary file).

3. Embed the Cult3D object in HTML pages, Adobe Acrobat and Microsoft Office documents and Macromedia Director®.

4. Apply for a publishing license.

Visit our web site at: www.cult3d.com

or contact: info@cycore.com

Cult3D software: Exporters, Designer and Viewers are free and available for download at: www.cult3d.com

The Cycore software that visualizes Cult3D objects, through browsers and other embedding software, are the Cult3D Viewers (plug-ins).

They are already installed in millions of computers on several platforms, like: Windows®, Mac®OS, Linux and others.

If a user visiting your web page does not have it already installed, the generated HTML code will guide him to the download of the Cult3D Viewer.

For CD-Rom presentations it is even possible to distribute the Cult3D software along with the Cult3D objects (please contact info@cycore.com).

Do not forget to pay a visit to the Cult3D community site, where enthusiast 3D artists and modellers share interesting info and amazing Cult3D objects: www.worldof3d.com

For the most updated information please visit our websites:

www.cult3d.com

Cult3D site

www.cycore.com

Cycore site

You are always welcome to contact us via e-mail to:

info@cycore.com

support@cycore.com

sales@cycore.com

System Requirements

Cult3D Designer System requirements:

	Minimum	Recommended
CPU	Pentium® 233 MHz or 100% compatible	Fastest possible Intel Pentium® with MMX or 100% compatible
RAM	64 MB	128 MB or more
Free Disc	20 MB	100 MB or more
O.S.	<ul style="list-style-type: none">- Windows 95/98/98 SE/ME- Windows NT 4/2000- Windows XP	

Software and Hardware Rendering

Cult3D visualization is performed by a powerful software rendering engine, which also has built-in support for hardware acceleration, when supported graphic cards are available in the system. During run-time, if requests exceed the graphic card capabilities (for example in terms of memory or supported features) the render will automatically switch to software.

In the Antialiasing and Bump Mapping chapters you will be able to read more about specific considerations related to software/hardware rendering.

To be able to utilize the hardware rendering, these are the requirements:

ATI graphic cards:

- ATI Radeon or higher
- DirectX 8
- ATI's latest driver that supports DirectX 8

NVIDIA graphic cards:

With Antialiasing

- NVIDIA GeForce or higher

- DirectX 8
- NVIDIA's latest driver that supports DirectX 8

Without Antialiasing:

- NVIDIA Riva TNT
- DirectX 8
- NVIDIA's latest driver that supports DirectX 8

Cult3D requires DirectX 8 for hardware acceleration, so it is important that your graphic card drivers support DirectX 8. If they don't, Cult3D will automatically use software rendering.

New Features Documentation

Cult3D Render

Antialiasing, new Progressive Mode

Antialiasing in computer graphics is used to alleviate artefacts (jaggies) induced by the discrete nature of the visualization screen.

With Antialiasing the rendered image exceeds, in terms of resolution, a certain number of times, called sub-pixel masks, commonly expressed in 2x2, 4x4 and so on. The final image is then obtained by scaling down the rendered image.

In Cult3D Anti-aliasing processing is driven by the following parameters:

- Antialiasing Mode.
- Antialiasing Delay (idle time in ms before the Antialiasing is triggered, only works with Automatic Antialiasing).

These parameters can be set in the HTML page (please refer to [Cult3D External Communication](#)) and/or with JavaScript methods.

Cult3D default settings of Antialiasing parameters are:

- Antialiasing Mode= 0
- Antialiasing Delay= 250 ms

Default settings are overridden by HTML specifications if present.

HTML specifications are overridden by JavaScript methods if present.

In Cult3D Designer version 5.3 the **Antialiasing Mode** parameter work in the following way:

- 0 : Automatic Antialiasing (with a sub-pixel mask equal to 5x5)
- 1 : Disable Antialiasing
- 2 : Enable Antialiasing (with a sub-pixel mask equal to 2x2)

The **Antialiasing Delay** parameter is taken into account only with the Automatic Antialiasing mode.

In this case Antialiasing is turned on after a certain elapsed time (specified by the Antialiasing delay setting) from:

- the end of any animation
- the end of any requested interaction (including any stated Friction duration)

Cult3D **Automatic Antialiasing** is now **progressive**: visual artefacts are reduced frame after frame thanks to the multi-pass render.

Progressive Antialiasing processing will be instantly aborted if any animation is requested.

The **Enable** setting generates a 2x2 Antialiasing at every frame, both in hardware and software rendering.

The **Disable** setting prevents any kind of Antialiasing to be applied.

The Designer uses the default Automatic Antialiasing setting.

Please note:

if you have an active particle system running in the scene and if the Antialiasing is on Automatic, no Antialiasing will be performed (in software rendering mode). That's because the particles are always moving.

Bump Mapping

In version 5.3 bump mapping is available both in software and hardware rendering (see the Software and Hardware chapter). All graphic cards supported by Cult3D are compatible with the implemented bump mapping method: DOT3.

The only exception is the NVIDIA Riva TNT, in this case Cult3D will simply enable the software render to handle the bump mapping in a proper way.

Bump mapping in version 5.3 only affects the diffuse texture/colour. Specular highlights will not be affected by the Bump map as in previous Cult3D versions, where the bump map was visible in specular areas, now it is only added on top. This means that your bump map will not be visible if your diffuse texture/colour is too dark.

This change has been done to support bump mapping in hardware and to make sure that it looks the same in both software and hardware rendering.

Cult3D Exporter

MaxScript Export to Cult3D

Using MaxScript the Cult3D exporter can export to *.c3d without the graphic user interface.

The MaxScript command for exporting files is:

```
exportFile "filename_string" [#noPrompt]
```

The [#noPrompt] parameter is optional and when set, no file dialogue window will be shown.

This is an example of how a MaxScript line could look like when exporting a scene to Cult3D without any file dialogs:

```
exportFile "teapot.c3d" #noPrompt
```

When using the command above default settings for the exporter will be used. In order to change the default export settings it is necessary to add custom properties to the max scene.

Custom properties are added in 3ds max™ through:

File->File Properties->Custom

Property name	Value	Description
CMSHAD	0,1,2,3	Type of shading 0 = Constant 1 = Flat 2 = Gouraud 3 = Phong
CMBiLi	0 or 1	Bilinear filtering 0 = Bilinear off 1 = Bilinear on
CTComp	0,1,2	Compression type 0 = 8 bit 1 = 24 bit 2 = Wavelet
CTQual	0-100	Wavelet level 0 = highest compression 100 lowest compression
CTUSize	1,2,4,8,16,32,64,128,512,1024,2048	Max size of X of texture
CTVSize	1,2,4,8,16,32,64,128,512,1024,2048	Max size of Y of texture

CAMAnim	0,1	Matrix animation (whole scene) 0 = animation not exported 1 = animation exported
CAVAnim	0,1	Vertex animation (whole scene) 0 = animation not exported 1 = animation exported

All of the above properties can be added to separate materials as well.
The name of the material must follow the name of the property.
For example:

"CMShadMaterial1=0 ".

The *.c3d file will be saved in 3ds max™ default file export directory.

Settings in the exporter window have higher priority than set properties in 3DS Max. Therefore, if you change settings in the exporter window (you are not using #noPrompt), they will be the ones used during exporting, and will be the ones saved with the *.max file.

The properties in 3ds max will be saved in the *.max file if no changes are made in the exporter window.

Cult3D Designer

Cult3D Stereo for Objects

In this chapter you can read about:

- Cult3D stereo concept and terms
- Creation of 3D Stereo objects in 3 steps

The Cult3D Designer version 5.3 is offering new tools, which will lead you easily through the creation of 3D Stereo objects.

In the following introduction we will describe how to build Cult3D Stereo objects for standard PC monitors and digital projectors with a screen width up to 1.5 meters.

In case you are interested in special viewing configurations, for example cinema's screens, please contact support@cycore.com.

Please Note:

Cult3D Stereo for Objects is especially indicated for objects visualization. It is not recommended for walkthroughs.

Stereo terms used in this chapter

Stereo Window:

The Stereo Window is the XY plane situated at the zero point of the z-axis of the camera frame. It corresponds to the screen plane. When objects are moved to a positive value of the z-axis they will be placed in the front of the Stereo Window and visa versa.

Stereo Safe Frame:

It represents the screen plane (Stereo Window) in the virtual scene. Everything shown inside this frame will be visible on the screen in the finished project. If an object or part of an object is outside this frame it will be cut off.

During visualisation the object or parts of the object will reach out of the screen if placed in front of the Stereo Safe Frame and visa versa. The Stereo Safe Frame is shown in the workbench window as a frame consisting of thick red lines.

Monitor Viewer:

This is a special Cult3D camera you will need to select when you want to use Cult3D Stereo for Objects. The Monitor Viewer will visualize your object based on the Viewing Configuration parameters.

The use of this camera will determine the presence of the Stereo Safe Frame in the scene.

3D Stereo Space (3DSS):

This is the 3-dimensional space inside which you can move your object safely in all directions. Everything shown inside the 3D Stereo Space will be visible on the screen in your finished project. The 3D Stereo Space (3DSS) is shown in the workbench window as a number of frames consisting of thin red lines in front and behind of the Stereo Window.

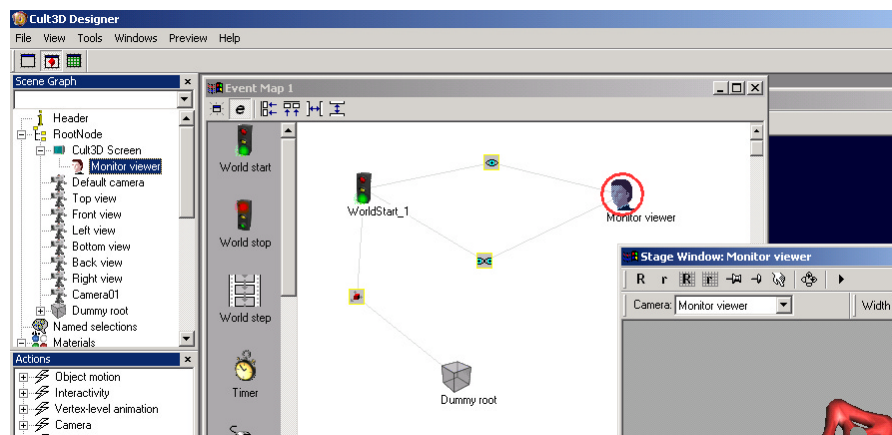
Meso stereo:

When approximately half of the object is in front of the Stereo Window (Stereo Safe Frame) and the other half is behind, it is called meso stereo. This gives you the best and most pleasant 3D Stereo effect you can obtain.

Creating a Cult3D Stereo Object

The two fundamental 3D stereo tools in the Designer are the Monitor Viewer and the Workbench.

In a few steps, or by using the Wizard, it is very fast to create a Cult3D Stereo object for a great 3D Stereo experience, without having to worry about 3D Stereo parameters settings.



Monitor Viewer

In order to create an impressive 3D Stereo object you will have to follow these 3 steps:

1. Set the size (in terms of pixels) of the desired visualization window and press the submit button.

2. Set the Monitor Viewer in 3D Stereo mode to be the active camera both in the Event Map and in the in the Stage Window.
3. Scale and position your 3D object. It must fit the Monitor Viewer and the 3D Stereo Safe Frame

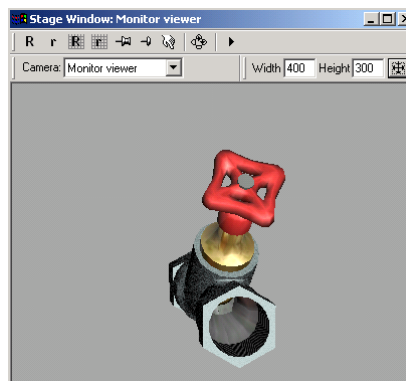
That's all!

Now we will describe the mentioned 3 steps in details.

(For the tutorial refer to *New 5.3 Samples\Stereo Tutorial* folder in your Cult3D Designer 5.3 installation directory.)

Step 1- Choose the Cult3D Window size

The stage window size must be set to the final Cult3D object visualization window size.



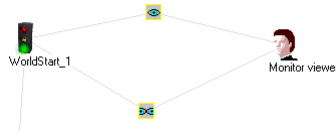
Window size setting

The HTML file, automatically generated with the Cult3D object file, will use this size setting for the dimension of the Cult3d window in the HTML page.

Step 2- Set Up the Stereo Tools in the Event Map

The Monitor Viewer must be selected and set to 3D Stereo mode by doing as follows:

1. Insert the Select Camera action in the Event Map window by dragging it from the actions menu. Then go to the Scene Graph\RootNode\Cult3D Screen and drag and drop the Monitor Viewer node in the Select Camera action.
2. Insert the 3D Stereo Features action in the Event Map window by dragging it from the Actions menu. Then drag and drop the above-mentioned Monitor Viewer in the 3D Stereo Features action.

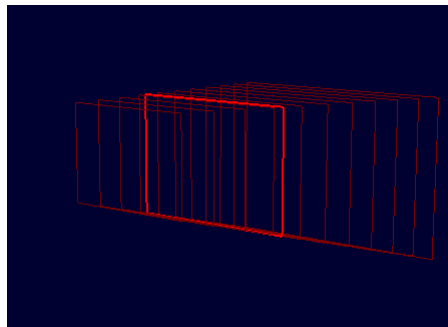


Monitor viewer and its actions as seen in the Event Map window

Step 3- Scale and Place the object

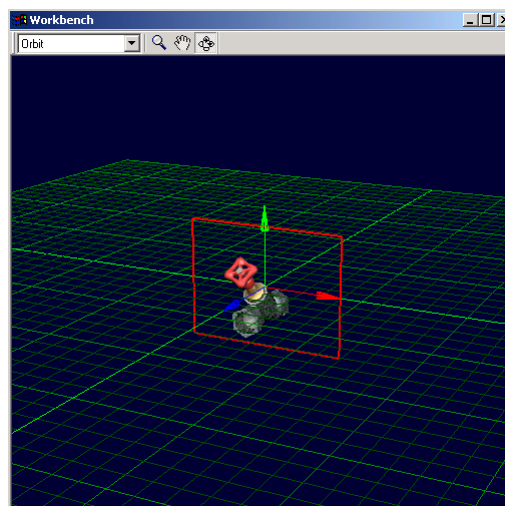
Scale and position your 3D Stereo object.

The goal of the third step is to locate and resize the object in such a way that it will fit the 3D Stereo Space: Cult3D Stereo for Objects takes care of the 3D stereo visualization of objects virtually located in a space/volume around the screen.



Stereo Safe Frame and 3DSS(3D Stereo Space)

1. Go to the Tools menu and select Resize/Center all.
This is placing the object in the world frame centre and resizing the object to fit the 3DSS.
2. Go to the View/Manipulator and fine-tune the size, rotation and placement of the 3D object that must never be larger than the Stereo Safe Frame.



Monitor Viewer and object as represented in the Workbench

Your object should be small enough to be able to pass through the Stereo Safe Frame (Stereo Window).

The best 3D Stereo effect is obtained by using meso stereo where half of the object is in front of the Stereo Window and the other half is behind.

In order to control the final stereo effect, check the position of the foremost part of your object with the 3D Safe Frame. You can set its parameters in the "Grid & Helpers" tab in the File/Preferences menu.

Before the fitting process the object could be far apart from the Stereo Window and/or too big to fit the 3DSS.

In the Workbench window you will be able to check the spatial relationship between the 3DSS and the 3D object.

When your object is properly located in the 3DSS, it will be visible in the Stage Window with the Monitor Viewer as the active camera.

Advice:

During the viewing process it is necessary to keep in mind that the Stereo Safe Frame (Stereo Window) must never cut off or touch that part of the object that reach out of the screen. Otherwise the 3D Stereo effect will be destroyed.

The Cult3D Stereo for Objects is dependent on a set of parameters, called Viewing Configuration, representing the relative position between screen and viewer user's eyes.

The default set of values will work more or less for all eyes positions respect standard PC monitors.

Just for your record, the given default configuration is approximately 50 cm from an average monitor (15" to 22") and the eyes approximately at the top border screen height.

If you are interested in special viewing configurations, such as big screens, cinema, please do not hesitate to contact Cult3D support (support@cycore.com).

The Monitor Viewer visualization dependency from the window position can be tested by moving around the Designer Stage Window or the Cult3D plug-in window: the 3D object perspective will change.

The wizard can automatically perform the three steps above.



Cult3D Wizard

Guidelines for using 3D Stereo:

Guidelines for 3D Stereo in general:

- The best 3D Stereo effect is obtained by using meso stereo, where half of the object is in front of the Stereo Window and the other half is behind.
- All 3-D viewers (stereo glasses) are causing light loss. It is therefore important to make the objects and backgrounds as bright as possible (without burning out the highlights) to compensate for the light loss.
- Pay attention to high contrast scenes and the use of black objects on a white background and vice versa, because it could cause "ghosting" problems.
- Objects with texture and shade usually work better than objects with uniform colours.
- Objects with lines or flat surfaces can cause "ghosting" problems.
- It is suggested to use a 3D Stereo background with a pattern or an image because it will act as a reference plane and thereby enhance the 3D Stereo effect.

- Use a calibrated monitor, if possible, with a colour temperature on 9300 K in a dim ambient lightning to get the best 3D Stereo experience.
- Keep your 3-D viewer (stereo glasses) clean.

Guidelines for using 3D Stereo backgrounds:

A background can be implemented in two different ways:

1. By using the Designer background action.

If you are using the ColorCode 3-D technology the background must be ColorCode 3-D encoded (read more in the [Appendix A](#)). You can either download one of the free ready-to-use 3D Stereo background textures from <http://www.3dstereo4you.com> or you can have your own 2D background encoded in ColorCode 3-D. Please contact: sbs@3dstereo4you.com.

2. By using a 3D panel in the scene.

You can attach a 2D background texture or image on the surface of a 3D panel, and then place it in the space behind your 3D object in the scene.

Guidelines for using 2D backgrounds:

- If your 3D object are always going to be in front of the Stereo Window, you can use any kind of background: uniform colour, gradient, texture or image (Designer Background action). However, it is still recommended to use a 3D Stereo background because it will enhance the 3D Stereo effect.
- If your 3D object or parts of it can be moved behind the Stereo Window during the visualization, the background must be a 3D Stereo background unless you are using a uniform colour or a gradient. Otherwise you will get an unpleasant conflict between two different depth cues. This happens because the background is positioned in the screen plane and the object or parts of it will break through the background.

Note: When you are using the ColorCode 3-D technology in connection with uniform colours and gradients you will need to pay attention to the chosen colours. Please make a test, and if necessary adjust the colours by using the Hue parameter to avoid chromatic sheen.

Guidelines for ColorCode™

- If your scene contains pure yellow or pure blue colours you will get chromatic sheen. To avoid that you can adjust these colours by using the Hue parameter.
- Keep your ColorCodeViewer clean.

Walkthrough and Collision Detection

This powerful feature facilitates the creation of walkthrough presentations where first person navigation is automatically performed.

Viewer users can now control a virtual person's navigation in your virtual environments using mouse and/or keys commands. The simulated movement modalities are: walking, strolling, running, turning, climbing stairs and obstacles up and down (depending on the obstacle's height), moving laterally, crouching, falling and more.

During a walkthrough and depending on the floor and the objects in the scene, collision detection filters all movements. Appropriate feedback is then returned to the viewer user. If no environmental restrictions have been detected, motion request is implemented. In case of small changes of slope or floor height such as stairs, fallen trunks, and/or ditches, the virtual person continues his/her trip following the altitude change. For example, in case of walls or high obstacles, his/her movements are inhibited and if confronted with a deep ditch, the virtual person falls down. All of the above walkthrough features can be part of your virtual environment just by dragging and dropping the *Mouse/Keyboard – Navigation* action in the Event Map and connecting it to a proper camera.

In particular, the following is performed:

- Collision detection
- Responsiveness to viewer user's change of direction
- Responsiveness to viewer user's change of walking speed and posture
- Responsiveness to viewer user's change of view orientation

A human character in Cult3D has the following dimensions:

- Standing: 0.3x0.3x1.8 meters
- Crouched: 0.3x0.3x0.9 meters

The virtual character can climb maximum 0.5 meters step.

A human walk in Cult3D is implementing the following speed values (accordingly to some researches):

- Normal walk: 1.34 m/s
- Run: 2.68 m/s
- Sneak: 0.67 m/s

With this feature the concept of first person navigation has been introduced. Becomes, therefore, fundamental an adequate choice of the camera Field Of View (FOV) that must be set in the modelling application.

One important issue in creating first person navigation experiences (virtual reality walkthroughs, games) is the choice of the camera Field Of View.

Often 90 degrees is the final choice. But one drawback is that it recreates a fisheye lens effect: everything at close range is unnaturally enlarged, while everything else is drastically shrunk.

For example, a nearby human character will look with a distorted huge head and very tiny feet.

Adopting FOV smaller than 90 degrees instead seems to give bad feelings to users navigating in close environments: the resulting view is limited, distances are badly judged and it is not always clear against what the user is colliding.

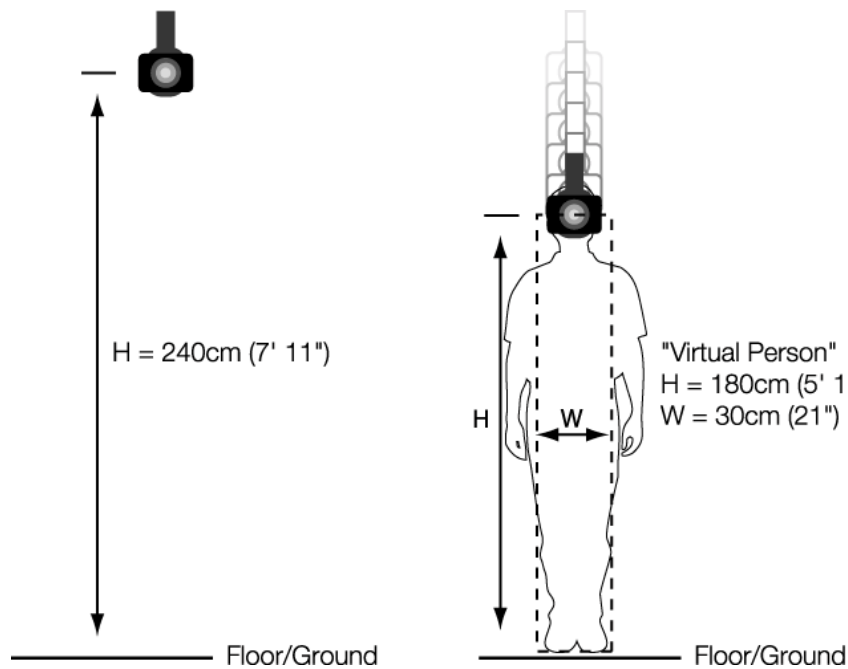
Besides the FOV value (that can be set from the modelling application), the above values cannot be changed.

Guideline for Modelling good Cult3D Walkthrough

There are few guidelines that can help you to avoid errors and maximize the Cult3D walkthrough experience.

- The camera connected to the Mouse/keyboard – Navigation action represents the virtual person point of view. Its position virtually corresponds to the head and 1.8 meters below are virtually positioned the virtual person feet. Therefore particular attention must be paid to the camera orientation, location and Field of View. During the explanation of the related requirements, two important keywords will be used: home grid and floor/ground. The home grid is the basic horizontal plane of the world coordinate system in the 3D scene, what you see from the top view camera. It is not a component of your model and it cannot have any slope, it is just a reference entity. Instead, floors and grounds are objects that are components of your modelled scene, they represent legitimate surfaces for walking, they can be more than one and may have different slopes.
 - The camera direction must be parallel to the home grid. In other words, the camera's vertical axes must be perpendicular to the home grid, regardless of the slope of the floor/ground beneath.
 - Place the camera where you want the virtual person to start its walkthrough.
 - The camera must be positioned above a floor/ground, at a height that is equal or higher than the value of the "Height" of the virtual person, that is 1.8 meters (5' 11"). Following this instruction will result in a consistent virtual person placement: there will be enough room for its initial standing position. It is wise, and actually recommended, to place the camera at a higher position than "Height". The result during runtime will

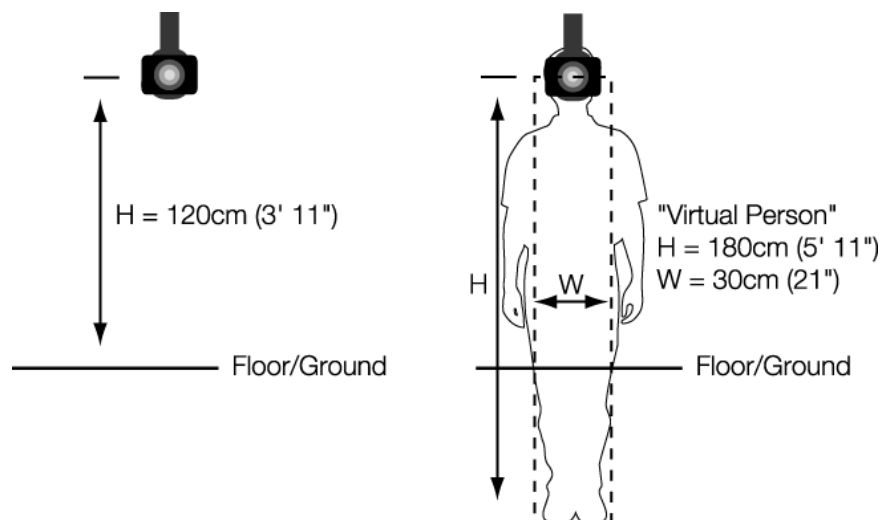
be that the virtual person will drop down to the ground/floor when the walkthrough begins (see example in following figure).



Camera in the modelling application.

Result during runtime.

If the distance between the camera and the beneath floor/ground is inferior to the "Height" (1.8 meters), the result in runtime will be that the virtual person will fall through the ground/floor, free falling out of the scene if no other floor is encountered.



Incorrect – camera placed too low compared to default values set.

A few examples:

- If the camera is at 3 meters from the floor, the "Virtual Person" will drop down for 1.2 meters until he will have touched with his feet the floor/ground below. His head will be then at 1.8 meters from the floor.
- If you have a camera 5 meters above the floor/ground, and below the camera you have a 2x2x2 meters box placed on the floor, at runtime, the virtual person will drop for 1.2 meters on the top of the box, ready to jump down the box and walk around.
- If your camera is at 3 meters above the floor/ground, and below the camera you have a 2x2x2 meters box placed on the floor, at runtime, the virtual person will drop for 1.2 meters on the floor, but inside the box.
- Which will result in the virtual person trapped in the box.

(for examples refer to the *New 5.3 Samples\New Navigation Sample* folder in your Cult3D Designer 5.3 installation directory).

It is wise, and actually recommended, to place the camera at a higher position than "Height". The result during runtime will be that the virtual person will drop down to the ground/floor when the walkthrough begins (see example in following figure).

- Increasing the Field Of View of the camera increases the portion of the scene that is visible. A good value for the Field of View of the camera is 90 degrees.
But too much of an increase could introduce perspective distortion, like fish-eye lens.
- Your architectural model must respect real scale dimensions. In this way, your virtual person will be proportionate.
It is much better if you model your scene in the appropriate scale. If that is not possible you can always use the Resize/Center All in the Tools drop down menu.
- Use Texture based lighting when possible.
Texture based lighting is a technique whereby a scene's lighting and material information are rendered as textures, these textures are then applied to the scene's surfaces. The advantage of texture based lighting is that it allows for the use of rendered shadows and radiosity effects, without overburdening the end-user system. When

you are using texture based lighting, it is convenient to use constant shading.

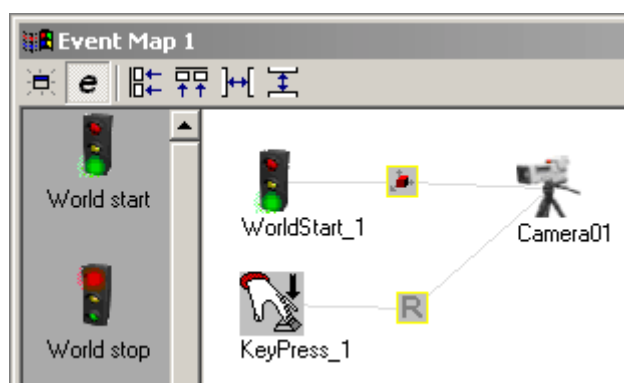
- The 3D scene must be suitable for walkthrough, possibly representing an architectural environments.
It is suggested that you add an exterior container (e.g. a box with sides facing inward) to your model to prevent the virtual person from dropping out of the scene.
However, at any time of the walkthrough, even during free falling caused by stepping out of the floor or through a hole, the walkthrough can be reset by pressing the Reset key, which will instantly place the virtual person back to the starting position in the walkthrough.

Creating a Walkthrough Project

Mouse/Keyboard – Navigation Action

Creating a walkthrough demo with collision detection can be achieved with the following few steps:

1. Create a new project in the Designer
2. Add the C3D file
3. In the Event Map, drag and drop in the World start event the Mouse/Keyboard Navigation action found in the list of Interactivity actions. Then drag and drop into the action icon the desired camera.
4. Check with a preview in the *Stage Window*.
Unless your camera is ill positioned or the environment's scale inappropriate, you should be able to navigate in the environment.
Refer to the guidelines for a correct scene set-up.
5. In order to be able to reset the scene when falling off the floor or just to be able to easily get back to the initial point, it is suggested to add the reset function to a *KeyPress* event.



6. When satisfied by the result, *Save internet file* from the File menu.

If you encounter problems you can eventually re-sync with the C3D file (not forgetting to resize after syncing if it was resized in the Designer project). You can also try to remove the camera from the Event map and add it again.

Navigating

Keyboard Commands

List of associations between functionalities and keyboard keys, eventually with alternatives:

Walk Forward	Arrow Up	w, W
Walk Backward	Arrow Down	s, S
Step Left	a, A	
Step Right	d, D	
Turn Left	Arrow Left	
Turn Right	Arrow Right	
Look Up	Page Up	
Look Down	Page Down	
Kneel	c, C	
Fast	z, Z	
Slow	x, X	

Please Note:

The disappearance of the scene from your view may be caused by free falling. The Reset key can then be used to instantly place the virtual person back to its starting position in the walkthrough.

MouseLook

By pressing the right mouse button you enter the MouseLook mode. In this case, walkthrough is controlled by all of the above keys and the mouse movement.

MouseLook	Right Mouse Button
Turn Left	Move Mouse at the left
Turn Right	Move Mouse at the right
Look Up	Move Mouse up
Look Down	Move Mouse down

For the last two functionalities, there is the possibility to switch to the Inverted MouseLook mode:

Look Up

Move Mouse down

Look Down

Move Mouse down

For the last two functionalities, there is the possibility to switch to the Inverted MouseLook mode:

Look Up

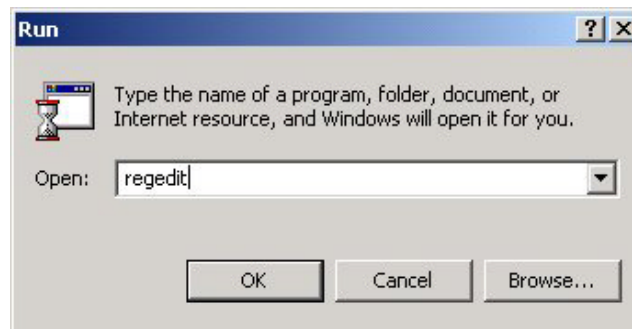
Move Mouse down

Look Down

Move Mouse up

In order to apply the inverted MouseLook you have to change the <invert mouselook> registry key:

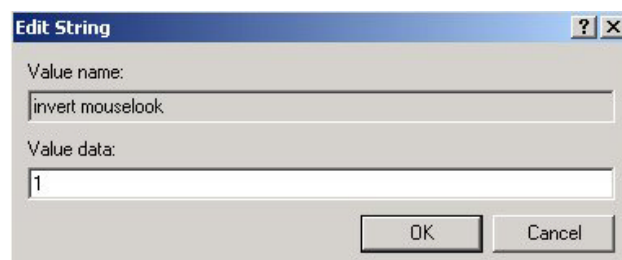
- Open the Run dialog box in the Start Windows Menu
- In the 'Open:' field edit 'regedit'
- Press the OK button or Return



- Open the folder in the registry at the following PATH:
- MyComputer\HKEY_CURRENT_USER\Software\Cycore\Cult3D\Common\Invert mouselook
- Double-click on the key called: <Invert MouseLook>
- In the Edit String dialog box, the chosen mode should be entered in the Value Data field.

Possible values and their corresponding mode:

- | | |
|---|--------------------------|
| 0 | MouseLook mode (default) |
| 1 | Inverted MouseLook mode |



All the commands can be used in combination. In case two opposite commands are entered, like <Move Forward> and < Move Backward>, the second one prevails.

The default walking speed has a value of 1.34 m/s (3 mph). Pressing the key related to the <Fast> function, while keeping pressed for example a <Walk Forward> key, speeds up your virtual person to its double velocity; releasing the <Fast> key brings speed back to normal speed. In a similar way the <Slow> function slows down your virtual person to its half velocity.

Walking Details

With *Walk Forward* and *Walk Backward* the camera "walks" (translates) along its Z-axis, following the eventual slope of the floor by automatically translating along the Y-axis.

Translation along the X-axis of the camera can be requested with the *Step Left* and the *Step Right* commands.

The rotation around the camera's X-axis (Pitch) is performed with the *Look Up* and *Look Down* commands. The movement is limited between - 90 and +90 degrees, to simulate the neck's constraints. The camera orientation does not follow the eventual floor slope, just like in the human walking, where the view is always perpendicular to the vertical orientation of the body unless an explicit request to look up or down is done. The Y-axis of the camera is always perpendicular to the world's ground plane.

Rotation around the camera Y-axis (Yaw) is done via the *Turn Left* and *Turn Right* commands. These commands will also change the direction of the walking.

Rotation around the camera Z-axis (Roll) is not allowed, since the human brain is not able to perceive a tilted reality when walking with his feet attached to the ground.

The gait of the virtual person is a motion capture sequence that cannot be edited.

Extended Arcball Action

The Extended Arcball allows the user of the Designer to edit the way mouse movements will control the behaviour of the selected objects in the scene during runtime.

You will access its functionalities by:

- Dragging and dropping the *Mouse – extended* action from the Action/Interactivity list into an event in the Event Map.
- Dragging and dropping into its icon in the Event Map the objects you want to be controlled by it.

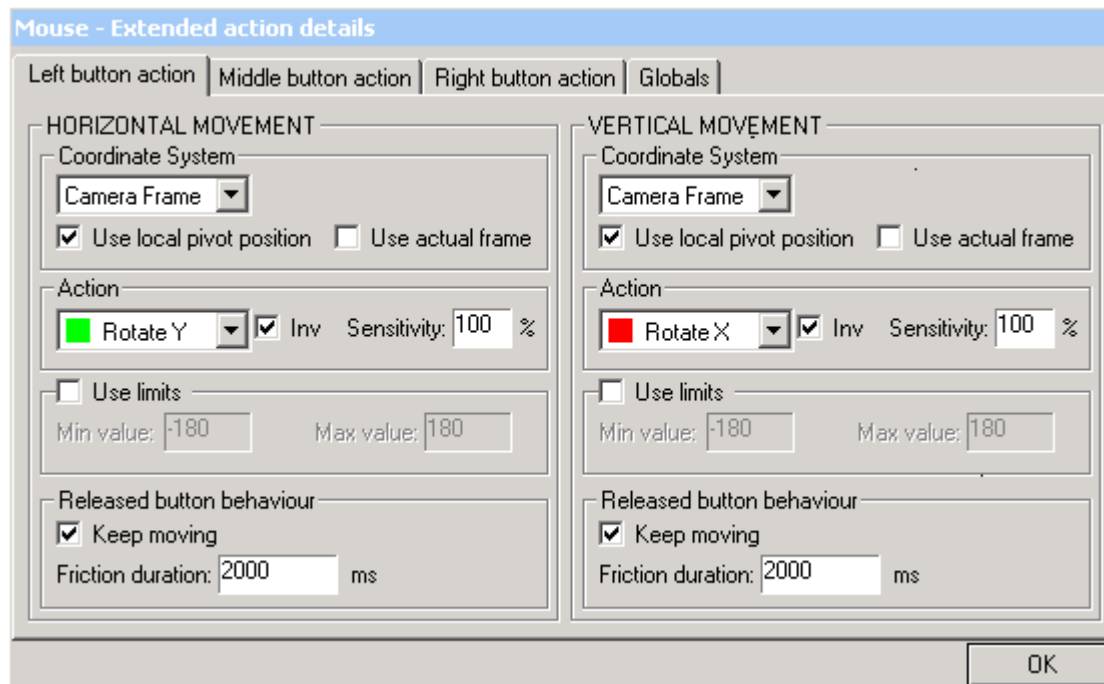
Double clicking the icon, or right clicking and selecting Details will open the *Mouse – Extended action details* window.

For every mouse's button, for every vertical or horizontal mouse movement, it is possible to set different parameters describing the motion induced by that specific controller channel.

During runtime all different channels are then combined in what will be the resulting arcball action applied to the connected objects.

Channels are always combined in the following order:

Rotation X, Rotation Y, Rotation Z, Translation X, Translation Y, Translation Z, Scale X, Scale Y, Scale Z.



Mouse – Extended action details window: mouse controller channels

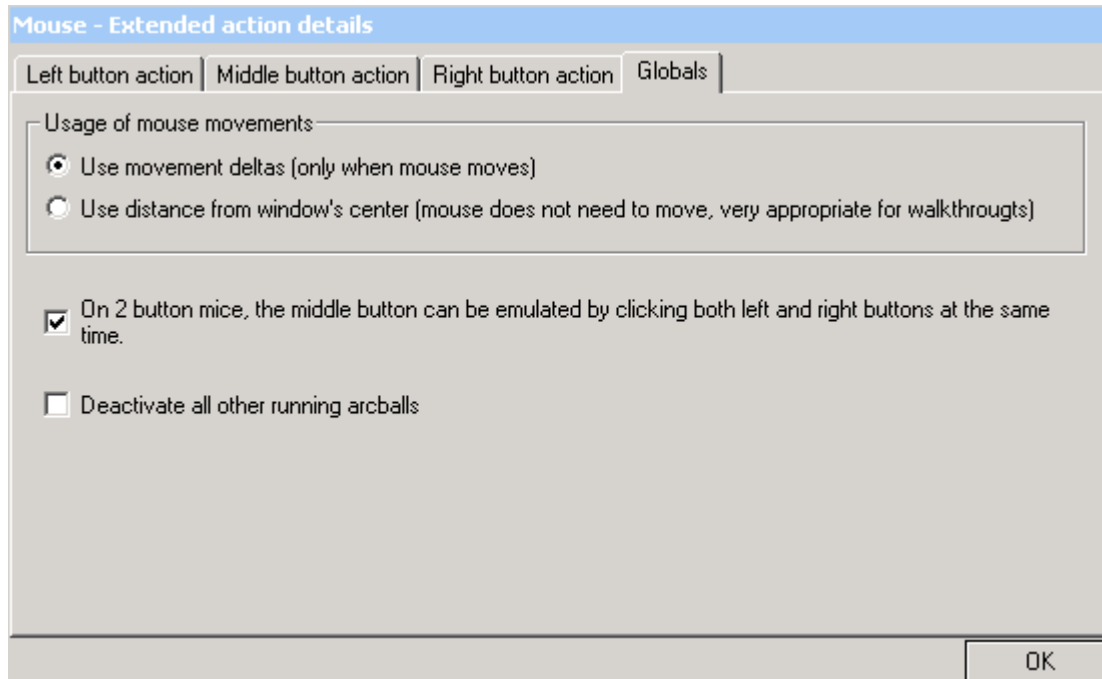
The interface for every channel of this controller has the same parameters that the standard arcball *Mouse – Arcball* has, with two more settings:

- The *Inv* check box in the Action section.
If checked, the resulting object motion will be inverted respect the mouse movement.
- The *Use limits* section.
By checking the check box, you can digit the desired limitations for the specific channel of the mouse controller.
Limits are relative to the state of the object when attached to the arcball.

The Extended Arcball can be used to create customized mouse interactions. The following is an interesting example:

in and out translation of the object respect the camera with a vertical mouse movement and rotation along the local Z axis of the object with horizontal mouse movements.

In the Globals tab, common settings for all channels are found.



Mouse – Extended action details window: Globals

In the *Usage of mouse movements* section a radio button allows you to choose between two different input modalities of the mouse movements:

- Use movement deltas
During the mouse movement, the two horizontal and vertical distances between the starting and the ending positions of the mouse are given as inputs to the action.
- Use distance from window's center
In this case the two input for the arcball action are the two distances (horizontal and vertical) from the mouse position in the window to the center of the window.
This means that it is possible to imprint motion just keeping the mouse still at any distance from the window's center.
If the mouse position is in the center, the inputs will be null: no motion.
This modality is particularly suggested for applications such as flythrough.

The last two check boxes are well described by their own names.

Tools menu

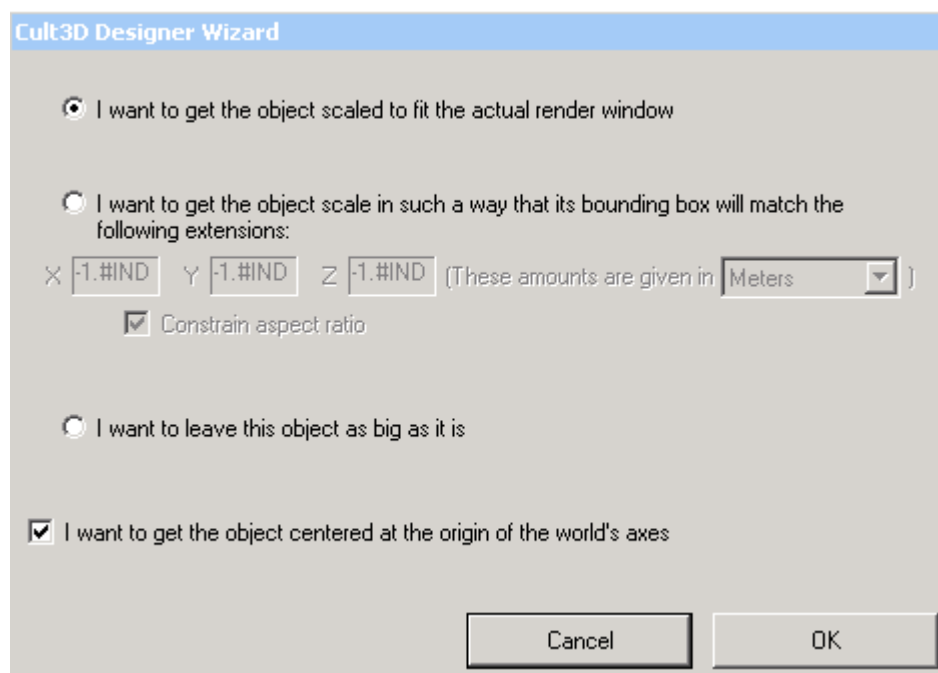
New tools are available in Cult3D Designer from the menu bar.

- Resize/Center All
- Hide and Unhide

Read more about them in the following sections.

Resize/Center All Tool

The new Tools dropdown menu in the menu bar presents a *Resize/Center All* tool, useful to rescale whole models and/or place them in the world's origin.



Resize/Center All tool

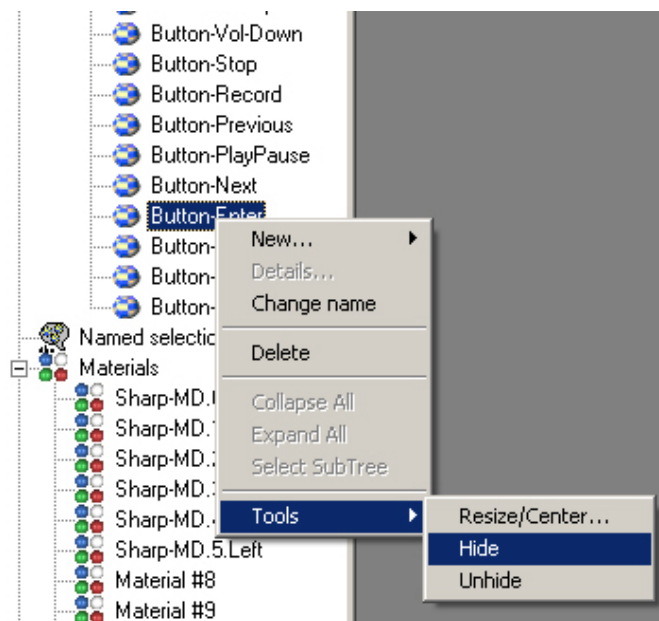
This tool can be very useful when loading *.c3d files: depending on how meshes have been modelled, they can be loaded very far away from the world's center and/or they can be extremely big or small in comparison with the units you defined for your projects. Activating this tool will get you to a good situation from where you can start your fine-tuning.

Hide and Unhide Tools

The dropdown menu in the Scene Graph has two new commands in the Tools selection: Hide and Unhide.

Hidden objects will not be shown until “Unhide action” will be called during the simulation.

Hidden objects are displayed in the Scene Graph with a faded icon.



Hide and Unhide Tools

The Workbench Window

The Workbench is a new extra tool that comes in handy when you are setting up your 3D scene.

Use the Workbench Window while editing the project and check the result in the Stage Window.

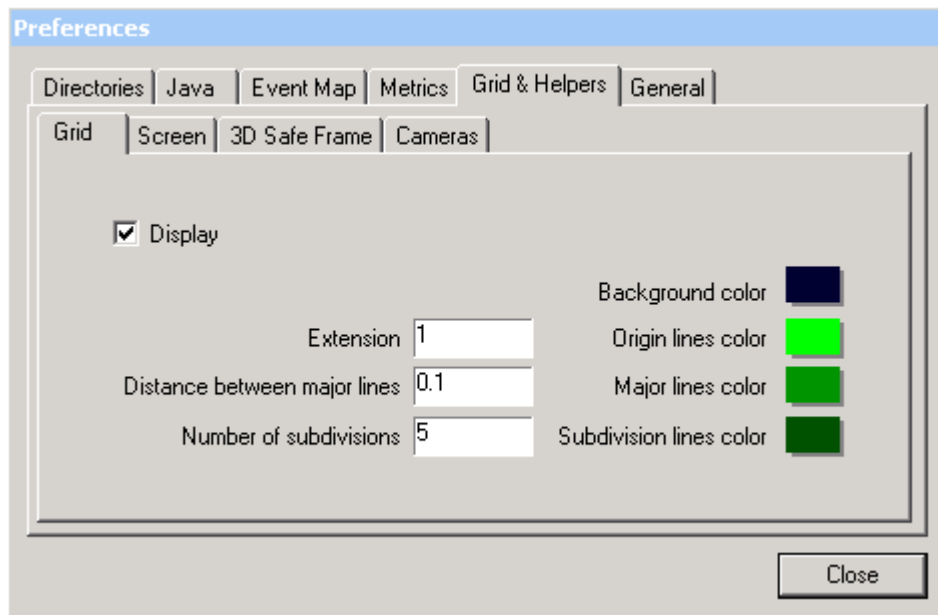
This Workbench window will give you a different perspective of your scene.



Select the new button in the menu bar to open the window.

The main element of this window is the Home Grid, a customisable wire frame plane, located in the centre of the world.

Its resolution and appearance can be modified in the File/Preferences/Grid & Helpers/Grid tab. Where you can also instruct the Designer to not visualize the grid.



Grid&Helpers tab in File/Preferences

In this window, local orientation axes of selected objects are displayed with the following colours association:

X: red
Y: green
Z: blue

There are other elements in this window designed to help you during the design of your Cult3D project: *Screen*, *3D Safe Frame*.

These tools are especially useful during the creation of 3D Stereo for Objects projects (please refer to Cult3D Stereo for Objects chapter). You can modify their appearance in the corresponding tabs in File/Preferences/Grid & Helpers. Where you can also instruct the Designer to not visualize them.

The Wizard

A new wizard has been introduced in the Designer to eventually facilitate your first steps in getting to know the Cult3D technology and/or to speed up some repetitive tasks.

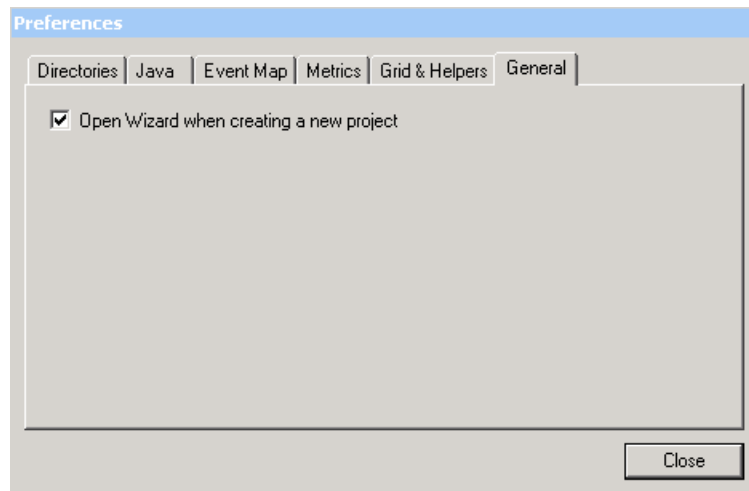


The Wizard

Requiring very little feedback, the Wizard will help you to create a simple "hello world" project, with *Select Camera* and *Mouse - Arcball* actions in a *World start* event.

The use of the Wizard is recommended for the creation of Cult3D Stereo for Objects projects. The Wizard can also convert existing projects to projects that implement Cult3D Stereo for Objects (please always consider the feasibility of the project in question, read more in the stereo documentation).

It is possible to avoid the automatic opening of the wizard by removing the check in the box available on the General tab of the File/Preferences.



General tab in File/preferences

Multiple Selection in the Scene Graph

Users can perform multiple selections of objects in the Scene Graph with the CTRL and SHIFT keys:

- CTRL key, after the first selection, hold down this key in order to add objects to your selection one at the time.
- SHIFT key, in order to select all objects included between the first selected object and the one selected holding down the SHIFT key.

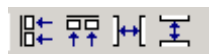
To select a whole sub-tree branch of the Scene Graph, right click on the root object and select *Select SubTree* from the dropdown menu.

To take away one object already selected, click on the object holding down the CTRL key.

Multiple selected objects can then be dragged into the Event Map.

Helpers in the Event Map

A new set of buttons on the event map frame helps the user to automatically align or even the space between the event map items.



You can find these buttons at the top of the Event Map window.

To align a group of items:

- Select all the items to be aligned.
- Keep the shift key pressed and click on the object to be used as reference.

- Push the “Align horizontally” or “Align vertically” button.

To even the spaces between items:

- Select all the items to be spaced.
- Keep the shift key pressed and click on the object to be used as reference.
- Push on the “Event horizontally” or “Event vertically” button. The maximum distance between the reference item and the two closest items to it will be used to space all the selected items.

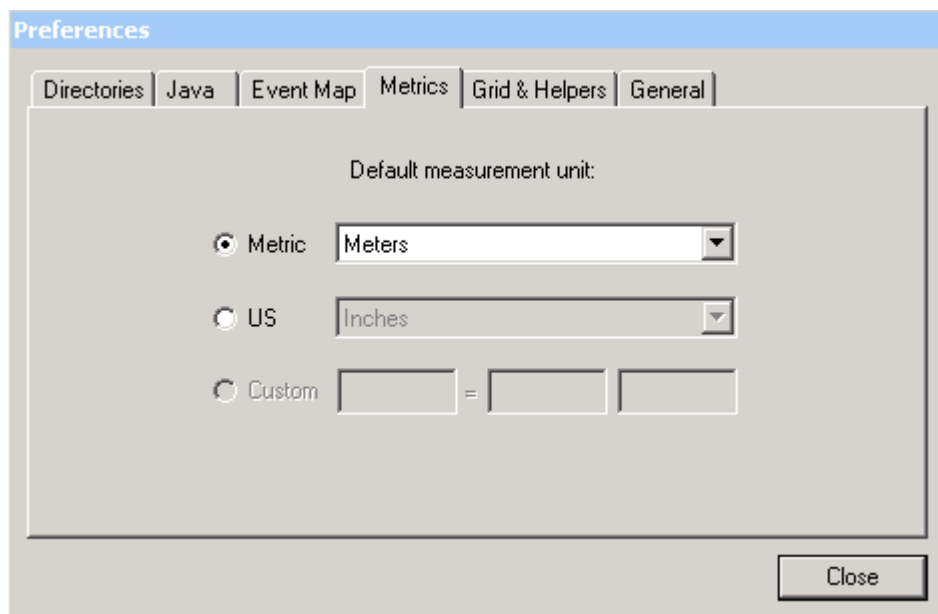
Metrics Tools in the Designer

In Cult3D version 5.3 the concept of measurement systems has been introduced.

Especially when working with the new walkthrough and 3D stereo features, it is very important to be consistent with real life scale.

Cult3D internal length measurement system is the Metric System, with the meter being its unit.

By selecting File/Preferences and the Metrics tab it is possible to change the units interface of distances and dimensions to the US system with inch as its unit.



Metrics tab in File/Preferences

By doing so you will be able to read and write amounts in the chosen unit in dialogs such as “Grid & Helpers” in the menu selection File/Preferences.

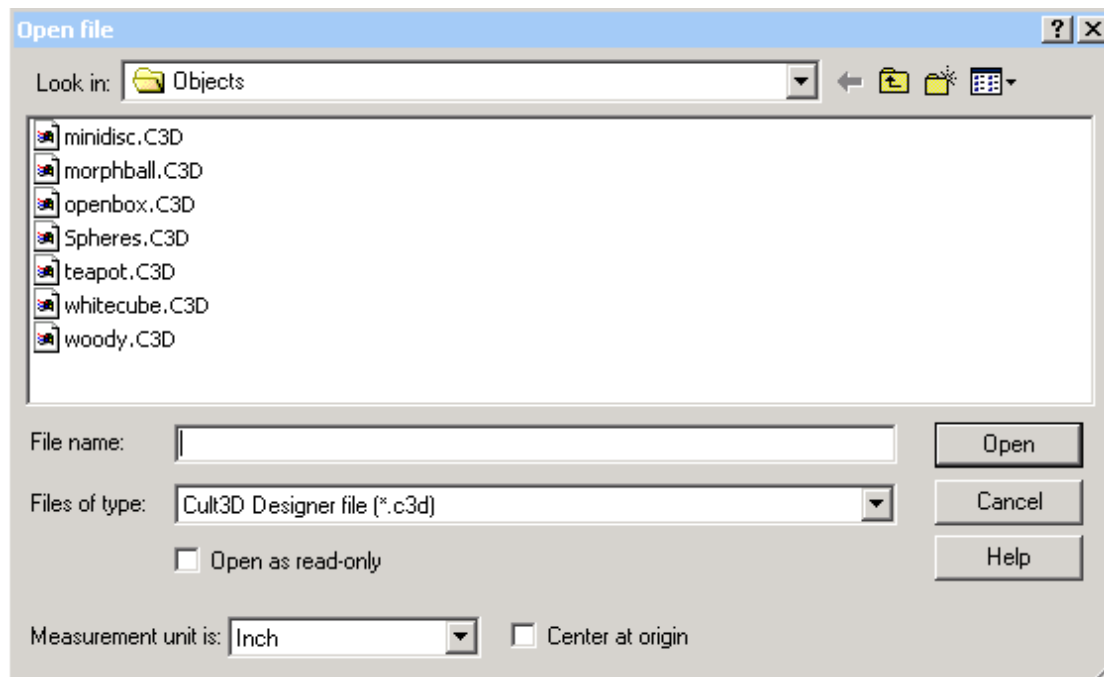
In Cult3D Designer, motion actions like Translation XYZ use arbitrary

units independent of the metrics.

Please note:

3dsmax internal system unit is always in inches, even when its interface has been differently configured.

A *.c3d file exported from 3dsmax and loaded into the Designer has lengths expressed in inches. Therefore you should also select inches in the "Measurement unit is:" combo box.



Opening a 3ds max file select inches in the "Measurement unit is:" combo box

The following table describes results of a unit movement in the Designer for all possible situations:

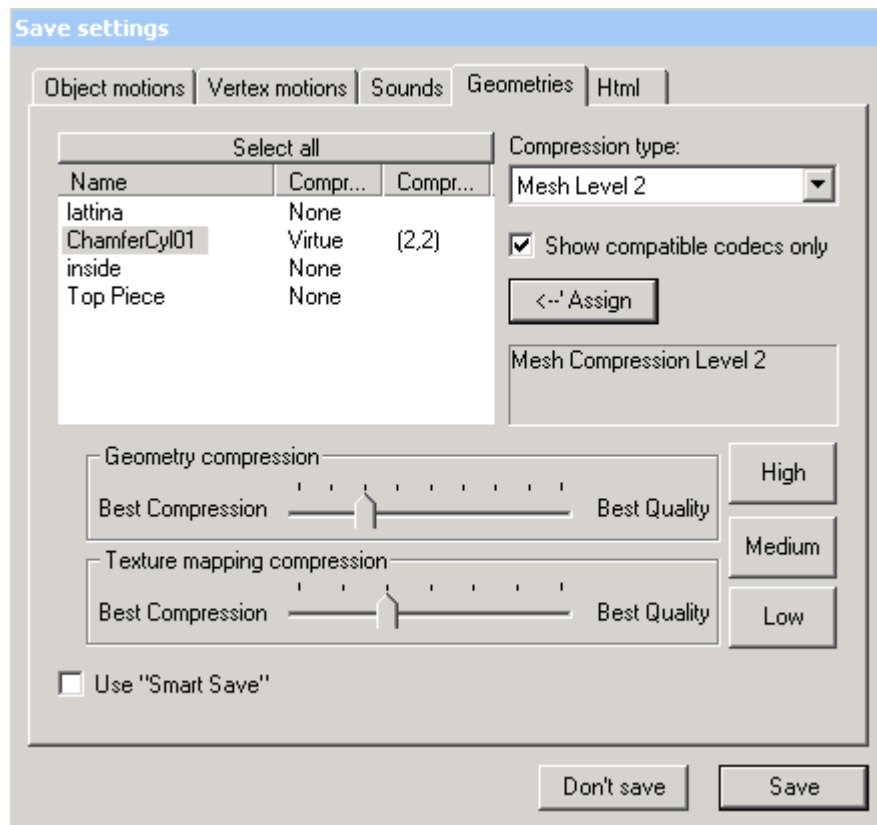
Unit and metric system used	Designer in meter (default)	Designer in inches (")
3dsmax in inches (default)	Object moves 1m or 39.37" (39.37 times the original movement in 3dsmax)	Object moves 1" (exactly the same movement as in 3dsmax)
3dsmax in meters	Object moves 1m (exactly the same movement as in 3dsmax)	Object moves 1" (1/39.37 the original movement in 3dsmax)

Where:

$$1 \text{ meter} = \frac{1}{39.3701''} \text{ and } 1'' = 0.02540 \text{ m}$$

Geometries Compression Settings

The new compression interface in the Cult3D Designer's Save dialog, for saving Internet files (CO files) presents two sliders and three buttons.



Save Internet File dialog: Geometries compression

The two sliders allow independent settings of the level of compression relative to:

- Geometry (mesh)
- Texture mapping (UV coordinates)

In this way more freedom is given during the research of a good compromise between visualization quality and file size.

In fact some models could be more sensitive to approximations of UV coordinates than to vertices coordinates and vice versa.

The three buttons correspond to three predefined levels of compression: low, medium and high.

Pushing one of these buttons will automatically set the status of the two sliders to the corresponding level of compression. The buttons can be used as starting points for the compression levels that can then be fine-tuned with the sliders.

Automatic HTML file creation

Together with the generation of the Cult3D object, during the saving process, the Designer now creates also an HTML file with very basic default settings.

This file, that allows a very quick visualization of the new Cult3D object, can function as a starting point for a later customisation of the HTML code.

The following is an example of the code (please refer to the Cult3D External Communication chapter for an explanation of the HTML code):

```
<HTML>
  <HEAD>
  </HEAD>

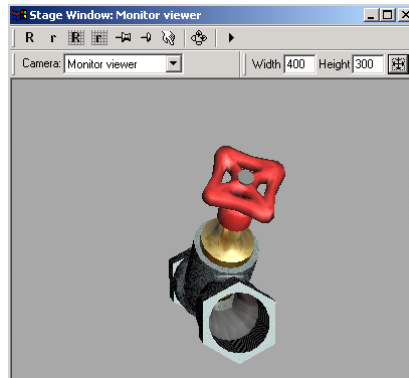
  <BODY>
    <OBJECT ID="CultObject"
      CLASSID="clsid: 31B7EB4E-8B4B-11D1-A789-00A0CC6651A8"
      CODEBASE="http://www.cult3d.com/download/cult.cab#version=5,3,0,2
      12"
      BORDER="0"
      WIDTH=400
      HEIGHT=300>
      <PARAM NAME="SRC" VALUE="LoadBottlesCaps.co">
      <PARAM NAME="ANTIALIASING" VALUE="0">
      <PARAM NAME="PBCOLOR" VALUE="FFFFFF">
      <PARAM NAME="VIEWFINISHED" VALUE="0">
      <PARAM NAME="DISABLEPB" VALUE="0">
      <PARAM NAME="ANTIALIASINGDELAY" VALUE="250">

      <EMBED NAME=CultObject
        PLUGINSOURCE="http://www.cult3d.com/newuser/index.html"
        TYPE="application/x-cult3d-object"
        SRC="LoadBottlesCaps.co"
        ANTIALIASING="0"
        WIDTH=400
        HEIGHT=300
        PBCOLOR="FFFFFF"
        BORDER="0"
        VIEWFINISHED="0"
        DISABLEPB="0"
        ANTIALIASINGDELAY="250"
```



```
</EMBED>
</OBJECT>
</BODY>
</HTML>
```

All parameters have default values, the only exception is represented by the WIDTH and HEIGHT parameters of the Cult3d window. These values can be different if the size of the window has been changed within the Designer.



Updates to the Cult3D Java API

The Java API has been extended with new Java classes and methods in order to better control Cult3D elements such as: mouse, camera, materials, events and tooltips (refer to the Java API documentation in the software package).

New and/or updated classes in the Cult3D 5.3 Java API.

Type	Name	Description
New class:	com.cult3d.device.CultWindow	Represents the window where the Cult3D scene is hosted.
New method:	getCultWindowDimension()	Gets the current size of the window hosting the Cult3D scene.
New method:	getNodeAtCoordinate(int x, int y)	Returns the topmost Node at the given screen coordinate, or null if there is none.
New method:	worldToWindow(float x, float y, float z)	Converts a world coordinate to a coordinate relative Cult3D Viewer window.
Updated class:	com.cult3d.world.Camera	Represents the active camera node.
New method:	getFarPlane()	Returns the far plane of the active camera.
New method:	getHorizontalFieldOfView()	Returns the horizontal field of view of the active camera.
New method:	getNearPlane()	Returns the near plane of the active camera.
New method:	getVerticalFieldOfView()	Returns the vertical field of view of the active camera.
Updated class	com.cult3d.Cult	A utility class.
New method:	sendMessageToHost(java.lang.String msg)	Sends a message to a Cult3D object embedded in the same HTML page.
Updated class	com.cult3d.world.CultEvent	Maps Manual Event for Cult3D Java.

updated/new method	public boolean trigger(boolean force)	Triggers the Manual Event this object represents. With this method you can choose if you want to trigger a deactivated event or not. This is equivalent to the "Trigger event" action in the Cult3D Designer.
Updated class:	com.cult3d.world.Material	Material management.
New method:	getTexture()	Gets the texture of a material.
New method:	getBumpTexture()	Gets the bump texture of a material.
New method:	getReflectionTexture()	Gets the reflection texture of a material.
New method:	getDiffuseIntensity()	Gets the diffuse intensity of a material.
New method:	setDiffuseIntensity()	Sets the diffuse intensity of a material.
New method:	setAmbientIntensity	Sets the ambient intensity of a material.
New method:	getAmbientIntensity	Gets the ambient intensity of a material.
New method:	setSpecularIntensity	Sets the specular intensity of a material.
New method:	getSpecularIntensity	Sets the specular intensity of a material.
Updated class:	com.cult3d.world.MouseDevice	Mouse management.
New method:	getMouseOverNode()	This method checks what node is under the current mouse position and returns the topmost node it finds.
New class:	com.cult3d.world.Tooltip	Provides a Java representation of a Cult3D Tooltip that exists in the scene graph of the Cult3D Designer.
New method:	addObject(CultObject object)	Adds the given CultObject to this Tooltips list over associated objects.
New method:	getAssociatedObjects()	Gets a list of all registered/associated objects for this Tooltip.
New method:	getFirstAssociatedObject()	Gets the first object that is registered/associated for this Tooltip.
New method:	getFixedPosition()	Gets the X Y coordinates where this Tooltip is in the Cult3D window.

New method:	<code>getNextAssociatedObject()</code>	Gets the next associated object for this Tooltip.
New method:	<code>getTexture()</code>	Gets the Tooltip texture.
New method:	<code>isActive()</code>	Checks if this Tooltip is active.
New method:	<code>isTrackingMouse()</code>	Checks whether this Tooltip is following the mouse.
New method:	<code>removeObject(CultObject object)</code>	Removes the given CultObject from this Tooltip list over associated objects.
New method:	<code>setActive(boolean state)</code>	Activates or deactivates this Tooltip depending on the value of the parameter state.
New method:	<code>setFixedPosition(int x, int y)</code>	Shows this Tooltip on a specific coordinate in the Cult3D window.
New method:	<code>setFixedPosition(Vector2 position)</code>	Shows this Tooltip on a specific coordinate in the Cult3D Window.
New method:	<code>trackMouse(boolean status)</code>	Activates or deactivates the option to let this Tooltip follow the mouse.

More Java documentation is found in the package downloaded together with the Cult3D Designer.

Please note:

In order to deal with materials and in particular sub-materials (one material can be compound by different sub-materials), it is strongly recommended to use the `com.cult3d.world.Texture` (managing both textures and colors) instead of the `com.cult3d.world.Material` class.

Cult3D.js file and JavaScript

A new version of the Cult3D.js file has been released.
It is easier now to instantiate Cult3D objects in layers/sub-layers.

Follow these steps to take full advantage of the improved Cult3D external communication capability:

- Use the newest version of the Cult3D.js file (see [Cult3D 5.3 Software Package](#)).
- Use the following guidelines to write the HTML code.

Please note:

When using the Cult3D.js file only one Cult3D object per each HTML page is supported.

Cult3D.js File

Cult3D.js code implements an interface between the Cult3D object and JavaScript that makes easier your JavaScript coding for communication with Cult3D objects.

The usage of Cult3D.js file is strongly suggested.

You do not need to change anything in the Cult3D.js file, just be sure to include it (with its correct relative path) in the head part of the HTML file (see below).

Suggested HTML code

The following is an example of an HTML code that correctly uses the new Cult3D.js file.

For your reference, number-labels have been added to every line.
The HTML example is also part of the [Cult3D 5.3 Software Package](#).

```
1. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
```

```

2. <html>
3. <head>
4. <title>Untitled Document</title>
5. <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
6. <script language="javascript" src="Cult3D.js"
type="text/javascript"></SCRIPT>
7. <script language="javascript" for="CultObject"
event="OnLoadFinished" type="text/javascript">
8. <!--
9. if (isActiveX())
10. {
11.     finishedLoading(this);
12. }
13. //-->
14. </SCRIPT>
15. <script language="JavaScript" type="text/javascript">
16. <!--
17. function MM_reloadPage(init) { //reloads the window if Nav4 resized
18. if (init==true) with (navigator) { if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
19.     document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
20. else if (innerWidth!=document.MM_pgW ||
innerHeight!=document.MM_pgH) location.reload();
21. }
22. MM_reloadPage(true);

23. var Cult3D;
24. var isLoaded = false;

25. function finishedLoading(refObject){
26.     if (typeof(Cult3D)=="object")
27.     {
28.         return;
29.     }
30.     Cult3D=new Cult3D_Object(refObject);
31.     if (Cult3D == null) {
32.         alert("not loaded");
33.     }

34.     if(Cult3D == null)
35.     {
36.         alert("Cult3D variable could not be initialized!");
37.     }
38.     else
39.     {
40.         isLoaded = true;
41.     }
42. }

43. function triggerCommand(arg) {

```

```

44.     if (isLoading == true) {
45.         if (Cult3D != null) {
46.             if (!Cult3D.triggerAction(arg,arg)) {
47.                 alert("Error trigger action");
48.             }
49.         } else {
50.             alert("Object not loaded");
51.         }
52.     } else {
53.         alert("Object not loaded");
54.     }
55.     Cult3D.setFocus();
56. }

57. // -->
58. </script>
59. </head>

60. <body bgcolor="#FFFFFF" text="#000000" onLoad="">
61. <div id="Layer9" style="position:absolute; left:78px; top:118px;
width:477px; height:400px; z-index:1">
62. <div id="Layer10" style="position:absolute; left:16px; top:14px;
width:441px; height:360px; z-index:2">
63. <div id="Layer11" style="position:absolute; left:4px; top:16px;
width:408px; height:324px; z-index:3">
64. <div id="Layer12" style="position:absolute; left:17px; top:14px;
width:383px; height:292px; z-index:4">

65. <object id="CultObject" classid="clsid:31B7EB4E-8B4B-11D1-
A789-00A0CC6651A8" width="342" height="274"
codebase="http://www.Cult3D.com/download/cult.cab#version=5,2,0,99"
>
66. <param name="SRC" value="TriggerEvent.co">
67. <param name="ANTIALIASING" value="2">
68. <embed name="CultEmbed" id="CultEmbed"
antialiasing="2"
69.         pluginspage="http://www.Cult3D.com/download/"
70.         width="342"
71.         height="274"
72.         src="TriggerEvent.co"
73.         onLoadFinished="finishedLoading('CultEmbed');"
74.         type="application/x-Cult3D-object">
75. </embed>
76. </object>
77. </div>
78. </div>
79. </div>
80. </div>

81. <a href="javascript:void(0);"
onClick="triggerCommand('StartRotation')">Trigger</a>

```

82.

83.

84. </body>
85. </html>

The **relevant details** of the code that you should focus on are:

- 6. The correct path to the Cult3D.js file must be given.
- 7. The parameter "for=" should have the same value as the "id=" parameter in line 65. (in this case is "CultObject" but it can be any string you prefer)
- 65. Read line 7. .
- 66. The parameter "value=" should be the correct path to the CO file (Cult3D Object).
- 68. The parameters "name=" and "id=" must have the same value (in this case is "CultEmbed" but it can be any string you prefer).
- 72. The parameter "src=" should be the same as the path you gave in the "value=" parameter in line 66.
- 73. The same above value (in this example string "CultEmbed") must be given as a parameter in the finishingLoading function.

Software Updates

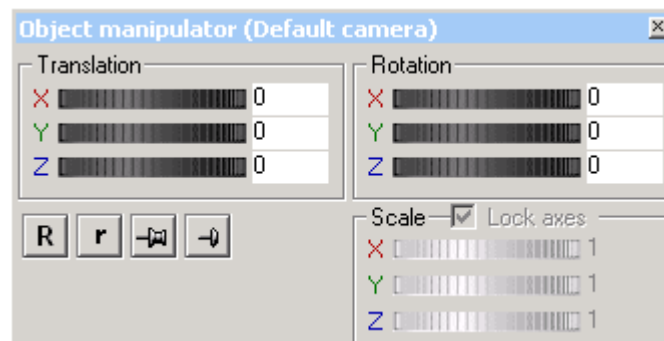
Toggle, Swap actions

The Toggle and Swap actions have been removed.

The same functionalities are obtained by using the Activate and Deactivate actions. Refer to our tutorial "Activate-Deactivate events" on the [World of 3D](#) web site (in the beginners section).

Manipulator

Whenever you have the necessity to reposition, change rotation or scale of any object in the scene, you can use the Manipulator.



Manipulator

It is activated from the View dropdown menu.

Select the object you want to modify, and fine-tune its parameters obtaining a simultaneous preview in the stage window and/or in the workbench.

The metric unit in which the Manipulator displays movements is set on the File/Preferences/Metrics tab.

Once you have obtained the desired situation, you can fix it by pressing the buttons available on the same tool window.



Manipulator button, it fixes all the objects in the scene to their current situations.



Manipulator button, it fixes the selected object to its current situation.

Instead, if you want to restore the last fixed situation prior fine-tuning you can use the Manipulator reset buttons.



Manipulator button, it resets the fine-tuning of all the objects in the scene.



Manipulator button, it resets the fine-tuning of the selected object.

Please Note:

It is not possible to restore the previous situation after a fixing button has been pressed.

Selecting Objects from Preview Window

In order to instantiate an object in the Event Map you can drag and drop the object name shown in the Scene Graph.

It is not longer possible to drag and drop the object from the preview window to the Event Map.

5.2 Features - Extended Documentation

In this part of the User Guide Supplement, you can find extra info related to features already described in the User Guide.

Worlds

In this chapter you will read about the Cult3D World feature:

- Meaning
- Applications
- Creation process
- Interface details

Worlds (sometimes referred to as sub-Worlds if contained in other Worlds) are Cult3D objects (*.co files) that can be dynamically loaded and merged into running Cult3D scenes.

Worlds can be used to create interactively evolving environments, some examples of possible applications:

1. An initial scene representing an empty room, where different furniture pieces can be loaded and moved around.
2. A basic car model to which different accessories can be applied.
3. An application for testing the look of different combinations of perfume bottles and caps.

As you can read above, the most common applications for the Worlds feature involve the possibility to load many different options in a fixed position of your scene.

A good way to implement these projects is to use dummy nodes in your modelling application to be used as place-holders in Cult3D (other possibilities are the use of Java or the *Manipulator* to place sub-Worlds). In the above case 3., you could create one dummy for the bottles, one dummy for the caps, and one main dummy containing the two previous dummies (the main Cult3D World will contain the three dummies, while sub-Worlds, to be loaded in the respective dummies, will contain single bottles and caps).

Attach, as children, all bottles to the first dummy and all caps to the second dummy, set their relative positions, and align their pivot points to their respective dummies.

Pay particular attention to the last step, objects and respective parent dummies must have the same pivot point position and axis orientation,

this point is fundamental for the correct placement in the final World of sub-Worlds (for examples refer to the *New 5.3 Samples\Load World Samples* folder in your Cult3D Designer 5.3 installation directory).

Then World entities must be created in the Designer.
sub-Worlds, objects loaded in other objects, are normal Cult3D objects.
sub-Worlds can contain other sub-Worlds, there is no limit to the nesting.

Worlds, Cult3D objects containing sub-Worlds, are normal Cult3D objects where the following settings must be done:

- Creation of the World entity in the Scene Graph.
- Definition of its parameters through its World details dialog box.
- Definition of its behaviour in the Event Map through World actions.

In the Scene Graph relative to the hosting scene, right-click on the *Worlds* item, select New and World from the dropdown menus.

A new sub-item named "World#" (where # will be the number of sub-items created so far) will appear in the "Worlds" item.

Double clicking on the new item will open its details dialog box, form where you can change the world's name (as for any other entity, you can rename the sub-item at your discretion), specify its local file name (the path and name of the CO file on your computer) and its URL path and file name which will be used once the whole project is published on the Internet. The path to the sub-World can be relative to its container, or absolute.

Clicking the "Do not allow this world to be instantiated more than" checkbox, will allow you to input in the nearby edit box the maximum number of times the World can be instantiated, since the same world can be loaded more than ones.

Now that the world entity has been created, you can use the World actions to set up its dynamic loading and unloading. You can find these actions under the group "World" in the actions box.

Available worlds-related operations are:

- Load World
- Unload World
- Hide/Unhide World

All actions contained in sub-Worlds can be activated as usual, exception done in two cases:

- Actions associated to Start World events
- Java actions

The only way to activate the two above actions is to call them by name from outside host, i.e. via JavaScript if the object is in a HTML page and to have all Java code in the main World.

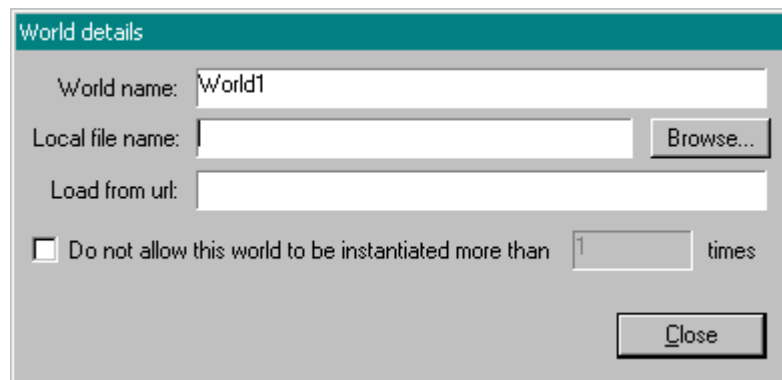
Example:

A Cult3D scene contains a node "cube" and a Load World action set to load "World.co" into "cube".

"World.co" is a Cult3D file with an event "Manual_event_1" connected to a rotate action on a node "sphere".

In the web page, besides the code to display the main scene (with the "cube " node), there can be a JavaScript calling the "Manual_event_1". Once the "World.co" file is loaded in the main scene, JavaScript can be executed, thus making the "sphere" rotate.

Details window

A screenshot of a 'World details' dialog box. It has a teal title bar. Inside, there are three text input fields: 'World name:' with 'World1' entered, 'Local file name:' which is empty, and 'Load from url:' which is empty. To the right of the 'Local file name' field is a 'Browse...' button. Below these fields is a checkbox labeled 'Do not allow this world to be instantiated more than' followed by a numeric input field containing '1' and the word 'times'. At the bottom right is a 'Close' button.

World name – Type the name of your world. It is the name shown in the Scene Graph.

Local file name - Type or browse for the local CO file corresponding to the world to be used by the Designer during the preview for testing purposes.

This parameter is ignored during the Cult3D viewer visualization.

Load from URL - Type the Internet URL of the CO file corresponding to the world to be loaded during the final Cult3D viewer visualization.

This parameter is the address from where the world will be downloaded by the published presentation.

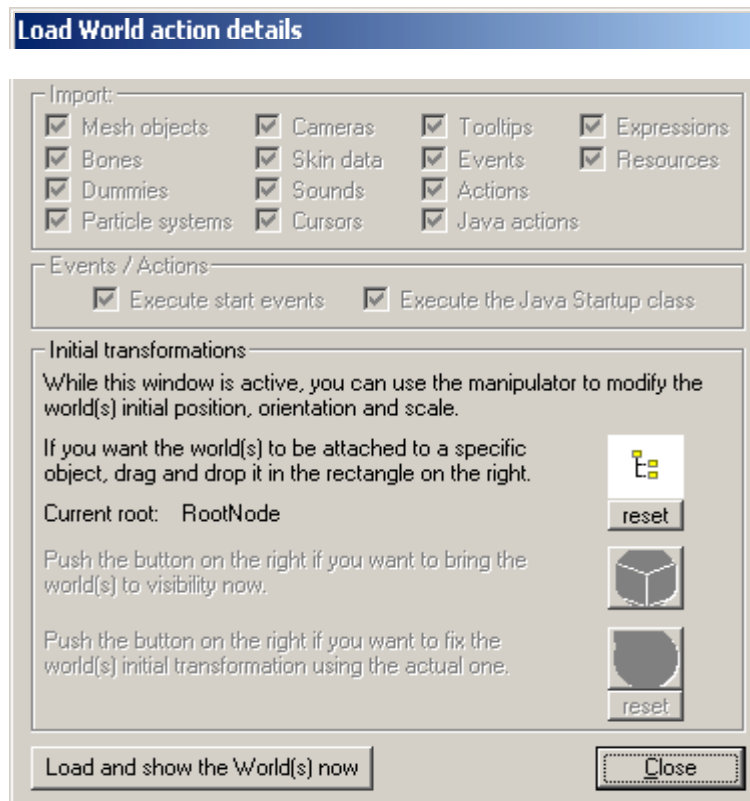
Do not allow this world to be instantiated more than – When checked allows the specification of the maximum number of times the World can be loaded in the main scene.

Actions

Load world Action

The purpose of this action is to load the world into the scene.

Drag the *Load world* action into an Event and drag the desired world into the *Load world* action in the Event Map or choose the world(s) from the action parameters list.



Load World Action details

Import - check what components of the world you want to download.

Events/Actions - if checked, the start event and Cult3D Java startup class will be executed when the world is loaded (remember that only the main Worlds support Cult3D Java).

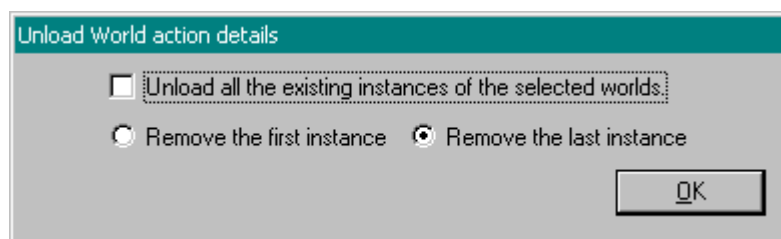
Initial transformations - settings for initial transformations of the world such as scale and position.

Unload world

This action unloads a world.

Drag the **Unload world** action to an Event.

Drag the defined world on to **Unload world** icon *or* choose world(s) from the actions parameters list.



Unload all existing instances of the selected world - if checked all clones of the select world will be unloaded from the simulation.

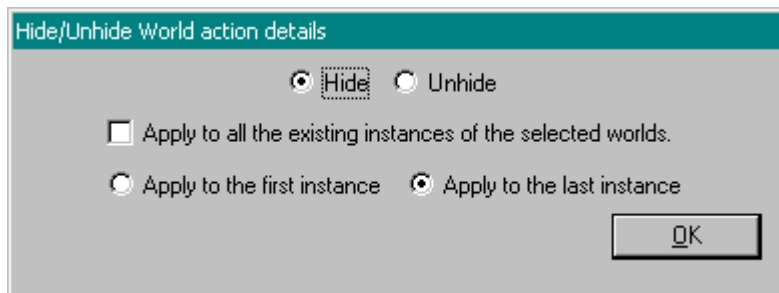
Remove first/last instance - only one world will be unloaded.

Hide/Unhide world

This action hides or unhides a world.

Drag the **Hide/Unhide world** action to an Event.

Drag the defined world on to **Hide/Unhide world** icon *or* choose world(s) from the actions parameters list.



Hide/Unhide - sets if the world is being hidden or unhidden.

Apply to all existing instances of the selected world - if checked all clones of the select world will be hidden/unhidden from the simulation.

Remove first/last instance - only one world will be hidden/unhidden.

Arcball Friction Duration

The Friction Duration determines the amount of time used by the animation engine for the transition of the rotation speed, from its speed at the release of the mouse to a speed equal to zero.

The whole Friction Duration will be observed no matter what is the initial speed of the animation: a very low initial value will result in imperceptible movements of the object, which could even look still to the viewer for almost all the Friction Duration time.

Please note:

The Antialiasing delay will always start its count down from the end of the Friction Duration, not from the apparent still state of the object.

Hotspots

HotSpots gives you the possibility to define a sensitive area in a texture.

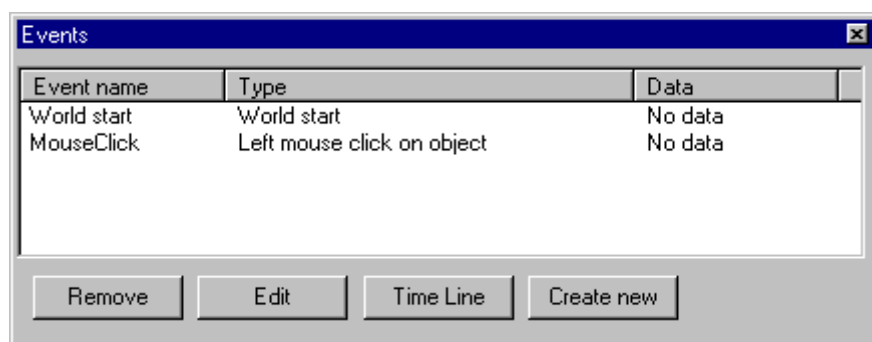
Even if the Cult3D Designer interface allows you to create more than one hotspot per texture, this practice is not recommended.

Events Windows

Events Tool Window

The Events tool window shows all events present in the scene. This tool allows an easy control and manipulation of existing events.

For example, in order to change an event type, you could remove the old event from the Event Map together with all its connections, create the new one and then connect the old connections to the new one, but the same result can be achieved with this tool in a much more efficient way.



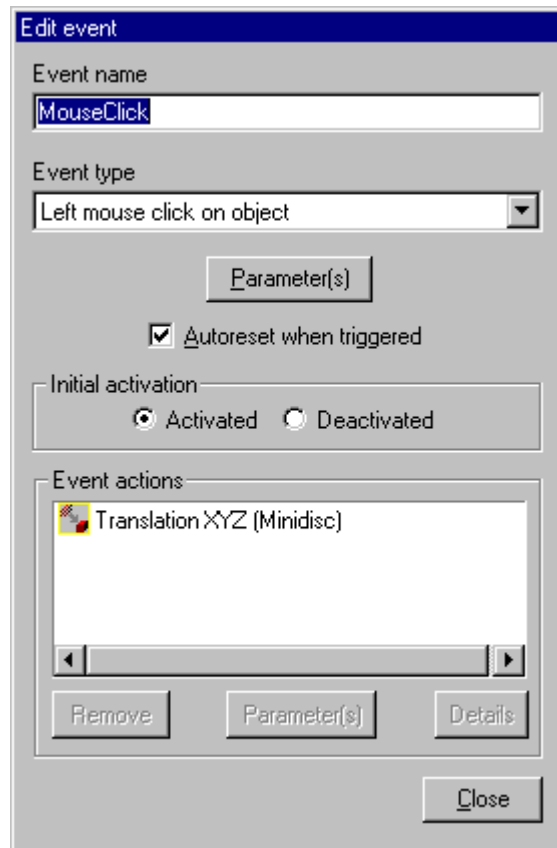
Events Tool window

Buttons in the Events tool window:

- | | |
|--------------|--|
| <Remove> | Deletes an Event. |
| <Edit> | Open the Edit event window to allow the editing of the selected event. |
| <TimeLine> | Open the selected event's TimeLine window |
| <Create new> | Open the Create new event window to allow the creation of a new event. |

Edit Event Dialog Box

This dialog box is opened clicking on the <Edit> button in the Events tool window or right clicking the relative event in the Event Map and selecting Edit.



Edit Event dialog box

The Edit Event dialog box is not only an easy way to edit events, but it also becomes very useful in complex projects, where following events and actions in the Event Map window becomes a challenge.

Details in the Edit Events dialog box:

<Event name>	Event name, as it appears in the Event map.
<Event type>	Allow you to change the type of event.
<Parameters>	Allow you to change the event input parameters.
<Autoreset when triggered>	When checked, instructs the event to be reset again right after it has been triggered.
<Initial activation>	Set the initial activation state.
<Event actions>	List all actions attached to the event.
<Remove>	Delete the selected action.
<Parameters>	Set parameters for the selected action.
<Details>	Set details for the selected action.
<Close>	Closes the Edit event dialog box.

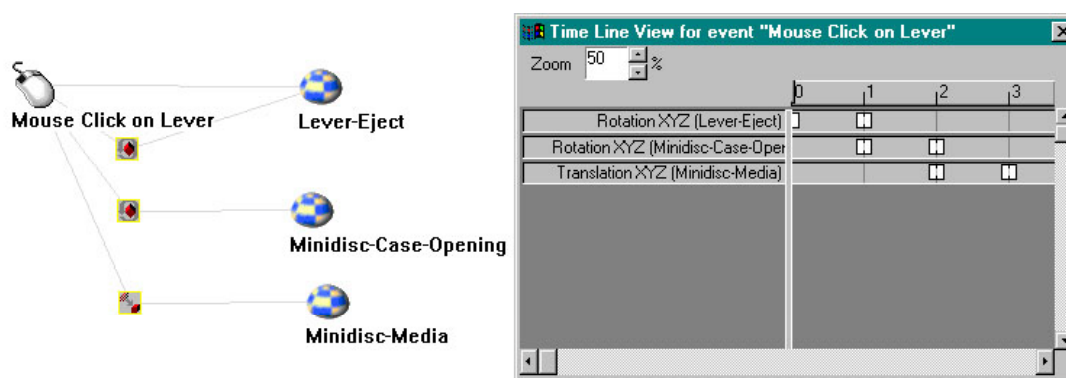
Time Line Tool

If you're familiar with the Track View in 3ds max™, or timelines in any animation application, you will recognize the use of the Time Line view in the Cult3D Designer. The difference between this Time Line and a standard timeline is that a Cult3D project has a separate timeline for every event. This means that you can have several timelines running at the same time.

The timeline is accessed by either right clicking on an event and choosing *Edit with Time Line View*, or by selecting an event in the Events Tool window and clicking on the *Time Line* button.

In the Zoom box in the top left corner, you can set the zoom level of the Time Line (higher value = more precise view over the time).

The picture shows the Time Line view for a mouse click event. In the left column of the window you see all actions attached to the event. Right clicking on actions is the same as right clicking on it in the Event map (Parameters, Detail and Delete). To the right, is the actual time in seconds, with "start" being when the Event is triggered. The white boxes on the right side show the start or stop of an action. In this case, the first rotation is located in the first second of the Time Line (0 to 1 second). It starts at 0 and stops at 1. By dragging the boxes to the left or right, you can change the time for start/stop and also the duration time of an action.



If an action is looped, three gray lines show it after the *stop* box. If the action is repeated a specified number of times, it is shown with faded white boxes, showing the time interval and the number of times the action is repeated.



The first action is looped and the second action is repeated three times.

The Time Line can replace many Timer events and sequence actions when building a project.

Note also that the Time Line window is read from top to bottom. That means that 2 events apparently starting exactly at the same time will actually be executed one after another (the first one being the one above the other one). The difference will be only microseconds but it will be there.

It's important to remember this fact if you execute an expression and test it against the value of a counter for example: one must be done before the other and the execution order is extremely important in that case.

Cult3D and Cult3D Java

Why Java in Cult3D

You can use the Cult3D Java API to meet your demands whenever you need special behaviour from Cult3D objects not directly provided by the Cult3D Designer.

Java is a cross-platform language: all your Java code will work for all Cult3D platforms.

For Java developers the Cult3D Java API is a powerful and customisable tool able to control many aspects of Cult3D objects.

Some examples are:

- Colours/textures
- Position and rotation
- Visibility
- Camera properties (such as field of view)
- Particle System

In few words, with Java you will be able to expand the Cult3D Designer action set and control Cult3D objects behaviour with the usual programming power: variables, statements, conditions, loops, etc.

Please note: Cult3D Java does not support any graphics libraries/APIs, such as AWT and Swing.

How to create Cult3D Java

Writing Cult3D Java code

For your Cult3D Java code you will have at your disposal the **Cult3D Java API**.

Following the directions for a correct **implementation** of a startup class and methods implementing bi-directional communication between Cult3D and Java will allow you to create the binding between your Java code and the Cult3D object.

Once you have your code in a *.java file, you will have to **compile** it in order to obtain the *.class file that you will have to **embed** in the Cult3D project.

Cult3D Designer only uses compiled Java code (*.class), it is not aware of Java source code (*.java files).

The compiled Java code (that can be in one or more *.class files) is finally encrypted in the Cult3D Internet file (*.co files).

Compiling Cult3D Java code

Once you have coded your Cult3D Java (*.java files) you have to compile it into Java classes (*.class files).

To be able to compile your Cult3D Java of course you must have a Java compiler.

We strongly recommend Sun's Java Compiler 1.1, which you can download at <http://java.sun.com>.

We also recommend you to read the Sun's Java tutorial: "The Java™ Tutorial, A practical guide to programmers", which you can find at the above URL.

Let's assume the following situation:

- Cult3DDesigner installed in "C:\Program Files\Cycore\Cult3D Designer\" folder
- Cult3DDevelop.jar in "C:\Program Files\Cycore\Cult3D Designer\" folder
- All your Java file(s) in "C:\project\foobar\" folder
- Java compiler installed in "C:\jdk1.1.8\" folder

In the following procedure it is described how to instruct the Java compiler about where to find the Cult3DDevelop.jar file (containing the Cult3D Java API classes and installed together with the Cult3D Designer) and how to compile your Cult3D Java classes.

- Open you Command Prompt
- Go to the directory where you have your Cult3D Java files (in our example "C:\project\foobar")
- Write the following command:

```
C:\jdk1.1.8\bin\javac.exe -classpath
".;c:\jdk1.1.8\lib\classes.zip;C:\Program Files\Cycore\Cult3D
Designer\Cult3DDevelop.jar" *.java
```

notice the -classpath parameter, it is there you tell the compiler where to find the Cult3Ddevelop.jar file.

If any error is found syntax errors and/or warnings will be issued.

Please note:

This is the only recommended way to compile Cult3D Java classes, don't try to add the Cult3DDevelop.jar file path to your system global classpath.

With the above command the compiled Java (*.class files) will end up in the folder(s) containing your Cult3D Java code (*.java files).

They will be ready to be included in your Cult3D project as described in the previous sections.

How to connect Cult3D and Cult3D Java

To embed your Java code in Cult3D you must implement the Cult3DScript interface, which will represent the interface between Cult3D and your Java code.

The following are the guidelines to be followed during the implementation of this fundamental class, also known as the **startup class**.

- The startup class definition must look like
public class YourStartupClass implements Cult3DScript
- The startup class must contain a public constructor with no arguments, even if it is not supposed to do anything.
- The Java startup class is the only class that can receive messages from Cult3D.
However you can send messages to Cult3D from any of your Cult3D Java classes.
- The startup class must also contain the public void cult3dDestroy() method. This method, called by the Cult3D Viewer just before closing, is where to place your cleanup and threads termination code.
public void cult3dDestroy()
{
//your clean-up code when Cult3D plug-in is about to be closed
}
- The startup class will also have to contain calls to other classes in case your project has several classes.
- The startup class must reside in the default package, whenever you use Java packages (read more in Sun's Java tutorial).

A startup class definition must look like the following:

```
// com.cult3d.Cult3DScript file must be included here
import com.cult3d.Cult3DScript;

//public startup class for your Cult3D Project which
//implements the interface com.cult3d.Cult3DScript
//(always mandatory)
public class YourStartupClass implements Cult3DScript
{
    // Your other classes declaration, if any
    private OtherClass x;

    // Standard constructor (always mandatory)
    public YourStartupClass()
```

```

{
    //instantiation of your classes if any
    x = new OtherClass();
}

//This method is called from the Cult3D Designer,
// and it forwards a call
// to the other class's method startObject()
public void startObject(String s)
{
    x.startObject();
}

// This method is called when the
// Cult3D viewer is being shutdown (always mandatory)
public void cult3dDestroy()
{
    //your cleaning code if any
    //your termination of your threads if any
}
}

```

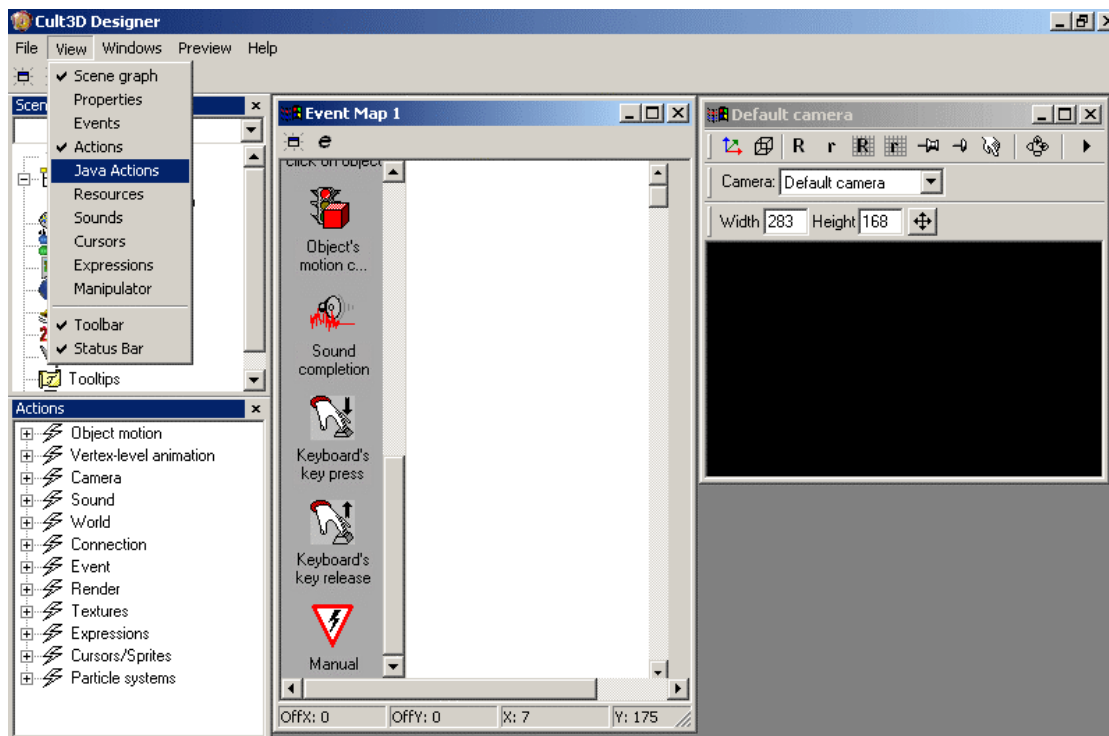
Where YourStartupClass can be any name you want.

In the following chapters you will read about the Java implementation of the communication between Cult3D and Java.
Now you will be able to read about this connection from the Designer point of view.

How to Embed Java in Cult3D Projects

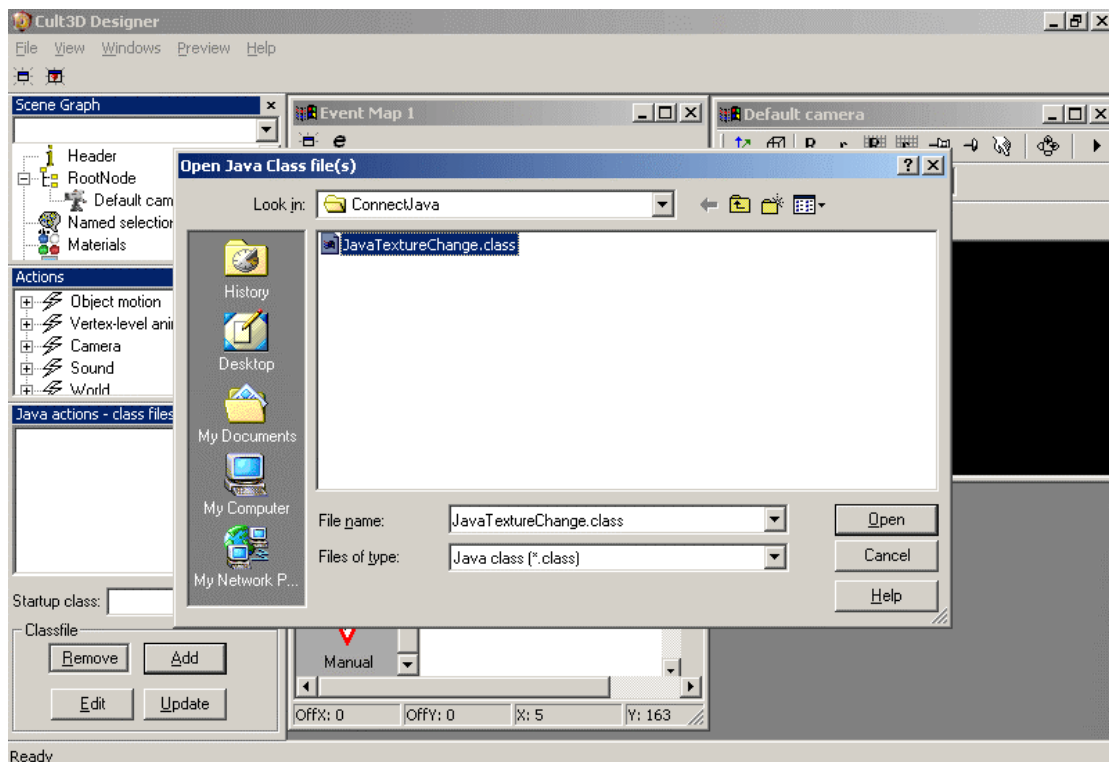
You must include all Java classes that you coded and compiled for your project in the Cult3D Designer Project.

Selecting from the menu bar View/Java Action will open the Java Action window.



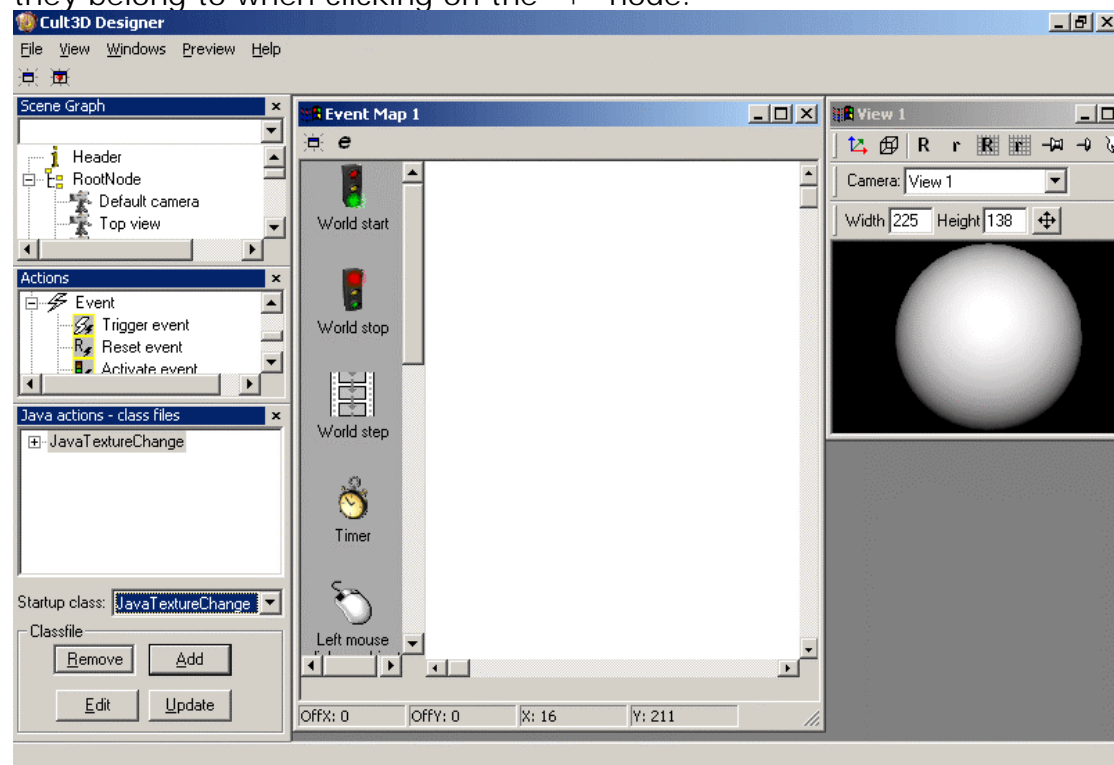
Select View → Java Action

By pressing the Add button in the Java Action window you will be able to select the *.class files that you want to embed in your Cult3D project, they can be in any folder you want.

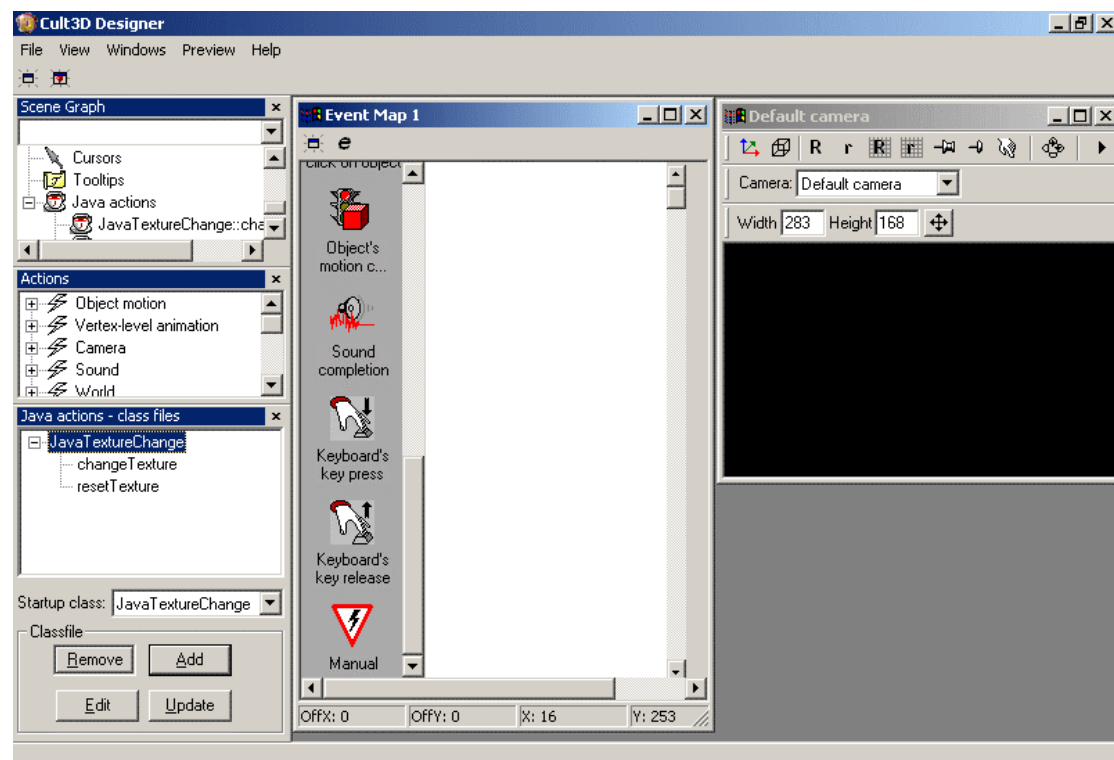


**.class files selection and embedding*

All the embedded *.class files will appear in the Java Action window.
All the methods observing the guidelines will also be displayed in the class they belong to when clicking on the "+" node.



All the embedded *.class files will appear in the Java Action window.



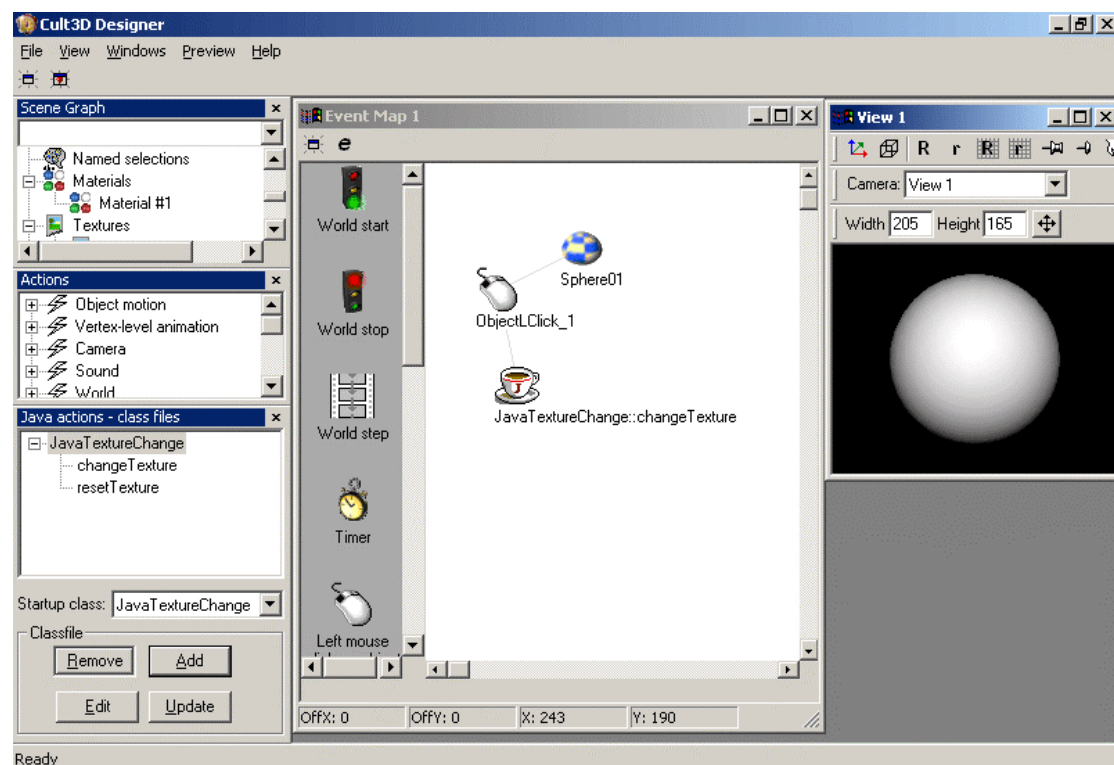
All the methods observing the guidelines will be also displayed.

Next step is to select the correct class as your startup class in the Startup class combo box.

Please notice that if you have a Cult3D Designer prior 5.2 you must write, in the Startup class text field, the name (with the extension .class) of your startup class.

In the Designer you can define a **call from Cult3D to a Java method** by drag&drop the method from the Java Action window in the desired event icon present the event map.

You must call only methods belonging to the startup class (using methods belonging to other classes is not supported, even if the Designer will let you drag&drop them in the Event Map).



Setting a call from Cult3D to Java

The preview of Java it is not supported in the Designer.

In order to test a Cult3D project containing Java, it is necessary to save the Internet file and test it in a Cult3D enhanced application.

Communication between Cult3D and Cult3D Java

Communication from Cult3D to Cult3D Java

The Java startup class is the only class that can contain methods for the reception of messages from Cult3D.

The following are the guidelines for the implementation of the message receiver methods:

- Public access
- Return void
- A single string as argument/parameter
- Resides in the startup class

These kinds of methods are the only eligible ones for inclusion in a Cult3D Designer Java action.

An example of how these methods should look like:

```
public void yourMethod(String designerArg)
{
    // Your custom code here
}
```

Where the names *yourMethod* and *designerArg* can be any valid Java names.

In the above example, if you connect the method *yourMethod* to a mouse event in the Event Map of the Cult3D Designer, during runtime when clicking an object connected to the mouse event with the mouse the name of the clicked object will be sent to yourMethod through the parameter string *designerArg*.

There can be any number of message receiver method in the startup class.

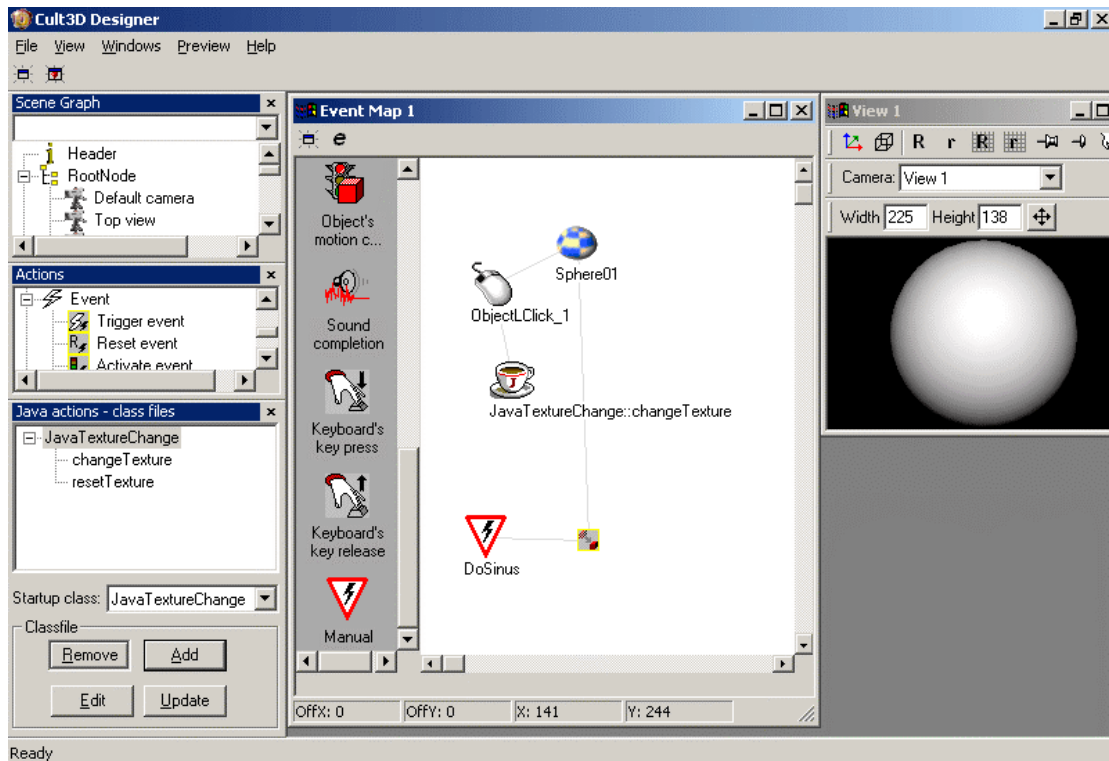
Communication from Java to Cult3D

Java can send messages to Cult3D with the following procedure.

Drag & drop a Manual Event in the Cult3D Designer event map. Change its name to something useful to you (right clicking on the manual event will open a pop-up menu where you will be able to select "Change Name" and open its dialog box).

Drag & drop the Cult3D Designer actions that you would like to run in response to the Java call.

The action/actions (you can have as many actions as you want run by the same event) should be dragged and dropped in the manual event that you just placed in the event map.



Manual Event named "DoSinus" will run the translation when triggered

The event map now contains all the necessary information for a call from Java to Cult3D that will run the object translation.

In Java in order to trigger the Manual Event ("DoSinus") you should instantiate a CultEvent with the name of the Manual Event (in this case "DoSinus") as a parameter to the constructor. The CultEvent class is found in the com.cult3d.world package. To actually trigger the DoSinus Manual Event you call the Cult3D Java method trigger() of your instantiated CultEvent. The instantiation of any CultEvent can be anywhere in any instantiated class (as long as it is legal Java syntax).

```
import com.cult3d.Cult3DScript;
import com.cult3d.world.CultEvent;
```

```
public class MyStartupClass implements Cult3DScript
{
    private CultEvent myEvent = new CultEvent("DoSinus");

    public MyStartupClass()
    {
        myEvent.trigger();
    }

    public void cult3dDestroy()
    {
    }
}
```

Cult3D External Communication

How the "JavaScript connection" between a Cult3D object and a containing HTML page works. There are a few points that we need to clarify.

Some implementations may be synchronous and/or have certain desirable timing characteristics; but the the following is what we guarantee, and what may be relied on.

- JavaScript and Synchronization

All JavaScript functionality is asynchronous.

Messages in either direction are guaranteed to arrive in the same order as they were sent; but there are no guarantees as to the timing/synchronization, neither with respect to messages going in the other direction, nor to the "process" sending the message.

So, e.g. if *triggerEvent* is called in a JavaScript in an HTML page, first on event "A" and then on event "B", events "A" and "B" are guaranteed to be triggered "A" first, "B" second; but they may be triggered on the same or different frames, and they may well be triggered long after the *triggerEvent* call returned in the JavaScript code. The same thing may happen in the opposite direction: if e.g. the 'send message to host' action is triggered on a specific frame, the message may well arrive to the page several frames later (real time).

- Sent messages to host are not queued

The *send message to host* function from a Cult3D object to an HTML page is not queued or synchronized in any way.

If more than one 'send message to host' is done from the Cult3D object to the HTML page (be it through a Cult3D action or a Java function call), there are no guarantees that the first message has been processed before the second one arrives, overwriting the first one.

- Cult3D events and timing

The "ONSCENELOADED" event in the HTML page should be used with caution. It is guaranteed to be triggered after any Java startup code has been executed in the Cult3D object, and before "ONLOADFINISHED".

But it may be called as early as before any *World start* event in the object has been triggered, as well as several minutes into the object's lifetime (i.e. long after *World start* has been triggered).

Appendix A: The ColorCode 3-D™

PowerAnaglyph

The worlds most advanced and flexible full colour 3-D Stereo system:

- Takes you to the leading edge of presentation
- Attracts full attention from your customer or audience
- Delivers the most powerful visual experience you can get

Advantages:

- Works on nearly all display media ranging from computer monitors (CRT – LCD – Plasma) and digital projectors to movies, offset- and inkjet prints
- Works with the same 3-D viewer on all media
- The 3-D Stereo image almost looks like an ordinary image when viewed without a 3-D viewer
- Works on standard equipment – platform independent
- High quality
- Easy to use
- Low cost

Description:

At first glance, the ColorCodeViewer™ with its amber-blue filters may remind you of the red-green glasses for anaglyphs. However, this state-of-the-art PowerAnaglyph™ technology is entirely different and the images are in full colour. In essence, the colour information is conveyed through the amber filter, and the parallax information - to perceive depth - is conveyed through the blue filter.

For further information about the ColorCode 3-D system please visit the following URL: <http://www.colorcode3d.com>

Trademark Acknowledgement

Microsoft, Microsoft Internet Explorer, Windows, Office and Word are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Java, AWT and Swing are trademark or a registered trademark of Sun Microsystems Inc. in the U.S. and other countries.

ColorCode 3-D, ColorCodeViewer, PowerAnaglyph are trademarks of ColorCode 3-D ApS.

Macromedia Director® is a trademark or registered trademark of Macromedia, Inc. in the United States and/or other countries.

Cult3D is a registered trademarks of Cycore AB.

Maya is a registered trademark of Silicon Graphics, Inc., exclusively used by Alias|Wavefront, a division of Silicon Graphics Limited.

Autodesk, Autodesk® VIZ™, Discreet 3ds max™, plasma™ are either registered trademarks or trademarks of Autodesk, Inc./Autodesk Canada Inc. in the USA and/or other countries.

Adobe, Acrobat are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Mac is trademark of Apple Computer, Inc., registered in the U.S. and other countries.

All other brand names, product names, or trademarks belong to their respective holders.