

Chat Mapper 1.2 Documentation

Table of Contents

Welcome	2
What's New?	3
License	4
User Interface	
Welcome Screen	6
Application Window	7
Configuring Layout	9
Conversation Tree	10
Application Preferences	11
Project Preferences	12
Assets	
Types of Assets	13
Asset Fields	14
Dialog Nodes	
Anatomy	15
Group Nodes	16
Adding/Removing	17
Linking	18
Configuring	19
Audio Files	20
Dialogue Markup	21
Scripting With Lua	
About Lua	22
Using Lua in Chat Mapper	24
Conditions and Scripts	26
Conversation Simulator	
Simulation Interface	27
Simulator Preferences	28
Exporting	
Basic Exporters	29
Technical Support	30

Welcome

What Is Chat Mapper?

Chat Mapper is a new high quality, easy to use tool for writing and testing non-linear dialogue, especially for video games and e-learning applications. You can use Chat Mapper to easily design conversations, write conditions and scripts using Lua, then test them for proper flow and comprehension.

What Is Non-Linear Dialogue?

Almost every video game that allows the player to interact with non-playable characters (NPCs) has a non-linear dialogue system where the player is allowed to choose what to say in a given situation and the NPC responds accordingly. The player's choices may affect the future decisions that the NPC makes and where the conversation leads. This allows the game to have branching story lines, which adds replay value and interest.

Traditionally this is done by creating a complex script using a word processor or screenplay development tool that is designed for linear story lines. Now, Chat Mapper provides an easy way to allow the writer to visualize the story and keep track of complex NPC behaviors.

Features

- Sleek and user-friendly graphical user interface
- Conversations visualized as a branching tree graph
- Define conditions using Lua for certain dialogue options to appear (e.g. never talked to this person, has a certain inventory item, etc.)
- Define how a dialogue selection affects the player-NPC relationship (e.g. the NPC may become more hostile in general if poor dialogue choices are selected, which affects future interactions)
- Link production files to dialogue entries (e.g. audio files, lip sync files, etc.)
- Automatically generate standard format scripts for voice actors
- Conversation simulation mode to test dialogue flow

System Requirements

Operating System: Microsoft Windows XP SP2; Microsoft Windows Vista; Microsoft Windows 7

Processor: 500 MHz Pentium (Minimum); 1GHz Pentium (Recommended)

RAM: 512 MB (Minimum); 1024 MB (Recommended)

Display Resolution: 1024 x 768 or Greater

What's New?

Version 1.2 (August 2011)

- All custom asset fields now show up in the add new dialogue window
- You can now type newlines in the dialogue text by pressing enter
- Improved emphasis tags including bold, italics, and underline options. Check out the new project settings.
- A global Lua function can now be defined in the project settings that is run only once during simulation.
- A Lua script can now be attached to a root node.
- New template system... when creating a new project, choose from predefined templates, or make your own.
- Improved welcome screen
- Chat Mapper Project Packager... export your project to a single package file including all media (Commercial license only).
- Improved XML export
 - Assets fields can be individually selected for XML export to reduce file size
 - New option to exclude fields with empty values
 - User variable types are now correct
- When validating a conversation, files are now checked for existence
- Fixed a crash bug when editing custom asset fields
- Fixed a bug when using Chat Mapper on a multi-monitor setup
- Changed fonts for better usability
- Numerous minor tweaks

Version 1.1 (January 2011)

- When a conversation is loaded, the view now starts centered on the root node
- MWV video playback is now supported for dialogue nodes
- A video browser was added to preview and add videos to dialogue nodes
- Zooming system was refined... zooming in and out no longer shifts the viewport
- Updated zoom keybindings: Ctrl+Plus now zooms in and Ctrl+Minus zooms out
- Default XML export encoding is now UTF-8 rather than UTF-16
- The window title now updates correctly on save
- Self-links are no longer allowed
- Enlarged size of node indicators and connector nodes
- Malformed dialogue links are now automatically repaired on project open
- Added undo/redo capability to Lua scripts and conditions editors
- Added syntax highlighting to Lua scripts and conditions editors
- Improved find function to limit results by various dialogue fields
- Added a "goto node" function accessible from the edit menu
- Added a replace function
- Added an autosave feature
- Fixed a crash bug that some users experienced on startup
- Fixed a crash bug when an actor in use is deleted and not replaced in a conversation
- Fixed a crash bug when closing the application while an audio file is playing in the simulator
- Fixed a bug where the actor and conversant labels on the root node would not update
- Fixed a bug with GetOutgoingLinks function with the exporter API

Version 1.0 (May 2010)

- Initial Release
-

License

URBAN BRAIN STUDIOS Chat Mapper End User License Agreement

THIS EULA SHALL APPLY ONLY TO THE SOFTWARE SUPPLIED BY URBAN BRAIN STUDIOS HEREWITH REGARDLESS OF WHETHER OTHER SOFTWARE IS REFERRED TO OR DESCRIBED HEREIN.

DEFINITIONS

(a) "Chat Mapper" and "Software" refers to Urban Brain Studios' Chat Mapper program, in each case, supplied by Urban Brain Studios herewith, and corresponding documentation, associated media, and online or electronic documentation.

(b) "Free Version" or "Freeware Version" or "Hobbyist Version" means a free version of the Software for personal use only, so identified, to be used only for non-profit projects. The Free Version is fully functional, but with some restrictions on the tools and export options available.

(c) "Indie Version" means a version which has been bought and an independent developer license has been provided by Urban Brain Studios.

(d) "Commercial Version" means a version which has been bought and a commercial developer license has been provided by Urban Brain Studios.

LIABILITY DISCLAIMER

THE CHAT MAPPER PROGRAM IS DISTRIBUTED "AS IS". NO WARRANTY OF ANY KIND IS EXPRESSED OR IMPLIED. YOU USE IT AT YOUR OWN RISK. NEITHER THE AUTHORS NOR URBAN BRAIN STUDIOS WILL BE LIABLE FOR DATA LOSS, DAMAGES AND LOSS OF PROFITS OR ANY OTHER KIND OF LOSS WHILE USING OR MISUSING THIS SOFTWARE.

RESTRICTIONS

You may not use, copy, emulate, clone, rent, lease, sell, modify, decompile, disassemble, otherwise reverse engineer, or transfer any version of the Software, or any subset of it, except as provided for in this agreement. Any such unauthorized use shall result in immediate and automatic termination of this license and may result in criminal and/or civil prosecution.

FOR CHAT MAPPER FREE VERSION ONLY

(a) Any project files or associated exported files generated by Chat Mapper Free Version MUST NOT be used for, or in relation with, any commercial or business purpose, whether "for profit" or "not for profit". Any work performed or produced as a result of use of this Software cannot be performed or produced for the benefit of other parties for a fee, compensation or any other reimbursement or remuneration.

(b) The Chat Mapper Free Version may be freely distributed, with exceptions noted below, provided the distribution package is not modified in ANY WAY.

(c) The Chat Mapper Free Version may not be distributed inside of any other software package without written permission of Urban Brain Studios.

(d) The Chat Mapper Free Version allows the user to publish its work according to the license agreement, but nor Urban Brain Studios nor any member of the company can be held liable for the content of the publication.

FOR CHAT MAPPER INDIE VERSION ONLY

(a) You may install and use the Software on a single computer; OR install and store the Software on a storage device, such as a network server, used only to install the Software on your other computers over an internal network, provided you have a license for each separate computer on which the Software is installed and run. A license for the Software may not be shared, installed or used concurrently on different computers.

(b) The Chat Mapper Indie version allows the registered user to publish its work according to the license agreement, but nor Urban Brain Studios nor any member of the company can be held liable for the content of the publication.

(c) The Chat Mapper Indie version guaranties to the registered user free updates for a whole version cycle and for at least 12 (twelve) months.

(d) The Chat Mapper Indie version may not be licensed and used by any companies or incorporated entities that had more than 10 (ten) employees or a turnover in excess of 250,000 USD in their last fiscal year.

FOR CHAT MAPPER COMMERCIAL VERSION ONLY

(a) You may install and use the Software on a single computer; OR install and store the Software on a storage device, such as a network server, used only to install the Software on your other computers over an internal network, provided you have a license for each separate computer on which the Software is installed and run. A license for the Software may not be shared, installed or used concurrently on different computers.

(b) The Chat Mapper Commercial version allows the registered user to publish its work according to the license agreement, but nor Urban Brain Studios nor any member of the company can be held liable for the content of the publication.

(c) The Chat Mapper Commercial version guaranties to the registered user free updates for a whole version cycle and for at least 12 (twelve) months.

(d) The Chat Mapper Commercial version may be licensed by any company, incorporated entity, or individual.

TERMS

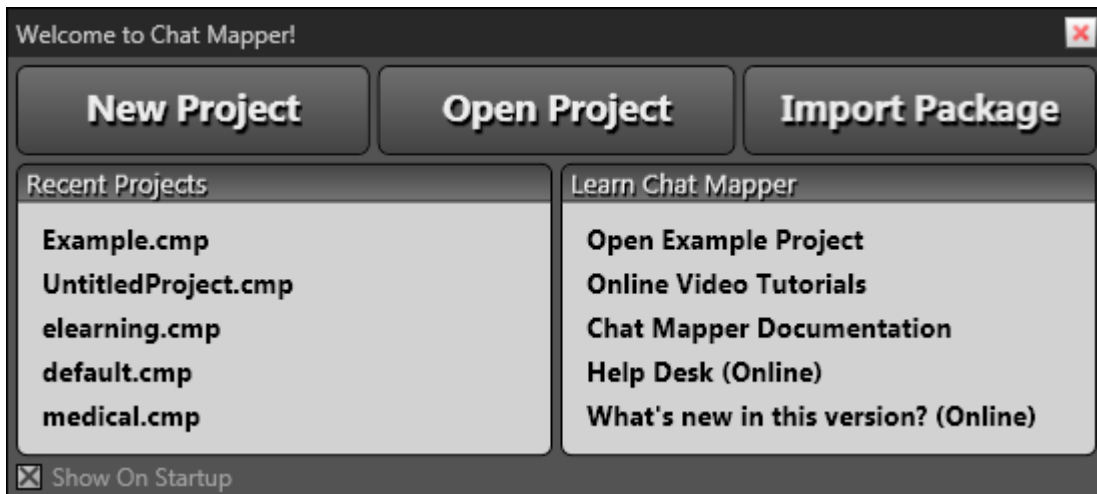
This license is effective until terminated. You may terminate it by destroying the program, the documentation and copies thereof. This license will also terminate if you fail to comply with any terms or conditions of this agreement. You agree upon such termination to destroy all copies of the program and of the documentation, or return them to the author.

OTHER RIGHTS AND RESTRICTIONS

All other rights and restrictions not specifically granted in this license are reserved by authors.

Welcome Screen

Upon running Chat Mapper, you are greeted with the Welcome Screen:



Welcome Screen

New Project

You will be prompted to save your new project right away because many operations depend on relative paths to the project file. When creating a new project, you are asked to choose a template as a starting point.

Open Project

Opens a project file on your hard drive that you have previously saved.

Import Package (Commercial License Only)

Imports a Chat Mapper Package (cmpkg) file including all media assets that the author used.

Recent Projects

Displays the five most recent projects that have been opened.

Learn Chat Mapper

Resources for getting the most out of Chat Mapper. The included example project makes use of many of the features of the software.

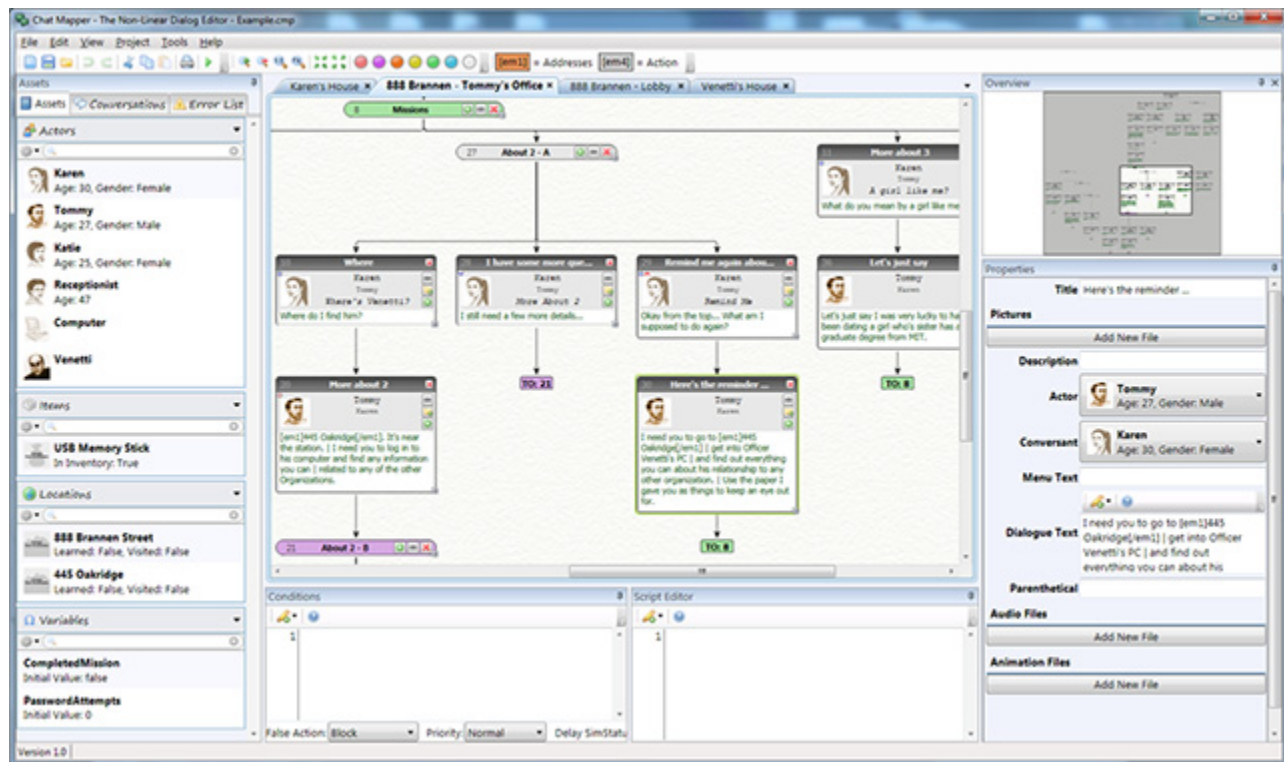
Show On Startup

Un-check this box if you would rather not display the welcome screen.

Application Window

Chat Mapper is easy to use and intuitive following current standards in Windows application development. In many cases, there are numerous ways to accomplish a task whether it be using the menu, tool bars, right-click context menus, or shortcut keys.

The window layout of the main Chat Mapper application is divided into several window panes as shown in the figure below.



Chat Mapper User Interface

Assets Browser

This is where you will find all of your assets except for conversations. You may sort and filter assets here using any of the asset fields. Each heading can be collapsed for easy navigation.

Conversation Browser

Here you will find all of your conversations. You may also sort and filter conversations.

Main Workspace

A conversation can be opened in the main workspace by double-clicking the conversation from the list. Each conversation is opened in its own tab.

Conditions Editor

Each dialogue node can have conditions associated with them. The conditions are written in Lua and will be described in detail later. Clicking a node in the tree will cause the conditions editor to display all conditions currently associated with that node.

Script Editor

Each dialogue node also has a Lua script that will run during simulation. The Lua script associated with the selected node is displayed here.

Properties Editor

All asset fields will appear here for editing. Properties of all assets and dialogue nodes are edited here.

Error List

Summary of all errors in your project.

Overview

Overview of the entire dialogue tree that provides a quick way to navigate around by clicking or dragging.

File Browser (not shown)

Shows all files on your hard drive and network. These files can be drag-and-dropped onto several locations in the application window.

Video Browser (not shown)

Shows a list of video files and embedded WMV clips in a selected directory. These can be dragged and dropped onto a node's video file attribute and will play during simulation.

Configuring Layout

Each tabbed window element of the interface can be dragged around in order to configure a layout that works best for you. Once you begin dragging a tab, you are presented with a graphical menu of locations to dock the tabbed window element.

By hovering your mouse over the various docking options while dragging, a preview is displayed of where the window will be docked. In addition, the overview and file browser windows can be floated or auto-hidden by dragging them, or pressing the small pin icon respectively. If you happen to close the overview or file browser and need to get them back, simply go to the **View | Show** menu.

Various pre-defined layouts have been saved and can be accessed by going to the **View | Window** Layout menu and selecting the desired layout.

Each time the application is closed, your last layout is saved and will be restored the next time you run Chat Mapper.

Conversation Tree

The conversation tree view is where you will spend most of your time editing your conversations so that they flow exactly the way you want them to. Some other features that we have not discussed is the ability to zoom in and out using the tool bar at the top of the main panel. There are also buttons to expand and collapse all nodes as well as to color the nodes. You may learn what each of these buttons do by simply hovering your mouse over the toolbar buttons.



Navigating The Tree

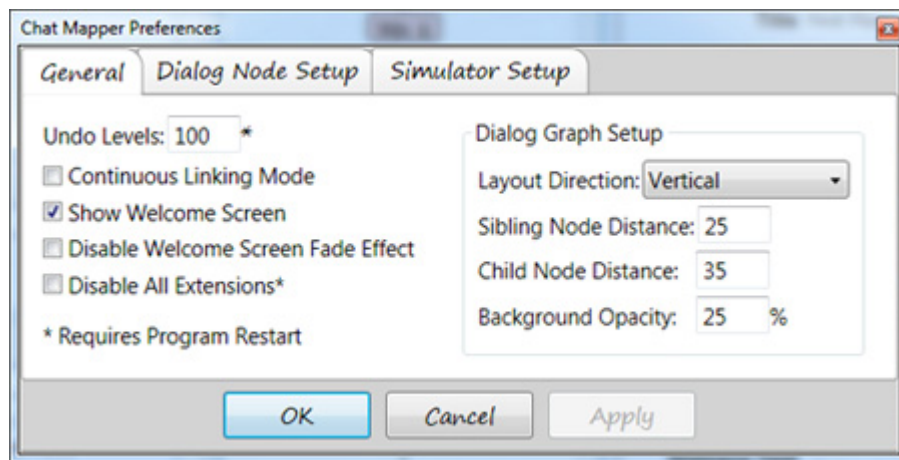
There are several ways in which you can navigate the tree to find and work with the individual nodes. You may scroll by either using the scroll bars on the side and bottom of the diagram, use your mouse scroll wheel to scroll vertically, click and drag the overview window, or middle-click the mouse and drag in the main workspace.

When your conversations trees start to get large, you may collapse and expand certain nodes by clicking the small [+] and [-] buttons in the bottom right corner of each node. Holding control while clicking will force all following nodes to expand or collapse.

If a dialogue node is currently selected, you may also use the arrow keys to navigate around the tree.

Application Preferences

Several application-level preferences can be set by selecting the **Tools | Preferences...** menu option.



Preferences Window

Undo Levels

Sets the number of operations that are stored on the undo stack. If you have low memory problems, you may want to try lowering this number.

Continuous Linking Mode

When enabled, allows for many dialogue links to be formed at once. Learn more about this in the help section about dialogue linking.

Show Welcome Screen

If you have previously disabled the welcome screen by un-checking the "Show On Startup" box on the welcome screen, you may turn it back on here.

Disable Welcome Screen Fade Effect

Turns off the fade-in animation when the welcome screen shows.

Disable All Extensions

Prevents the 3rd party exporters from loading when the application starts.

Layout Direction

Allows you to display the dialogue tree graph either vertically or horizontally.

Sibling Node Distance

Adjusts the distance between adjacent dialogue nodes on the same tree level.

Child Node Distance

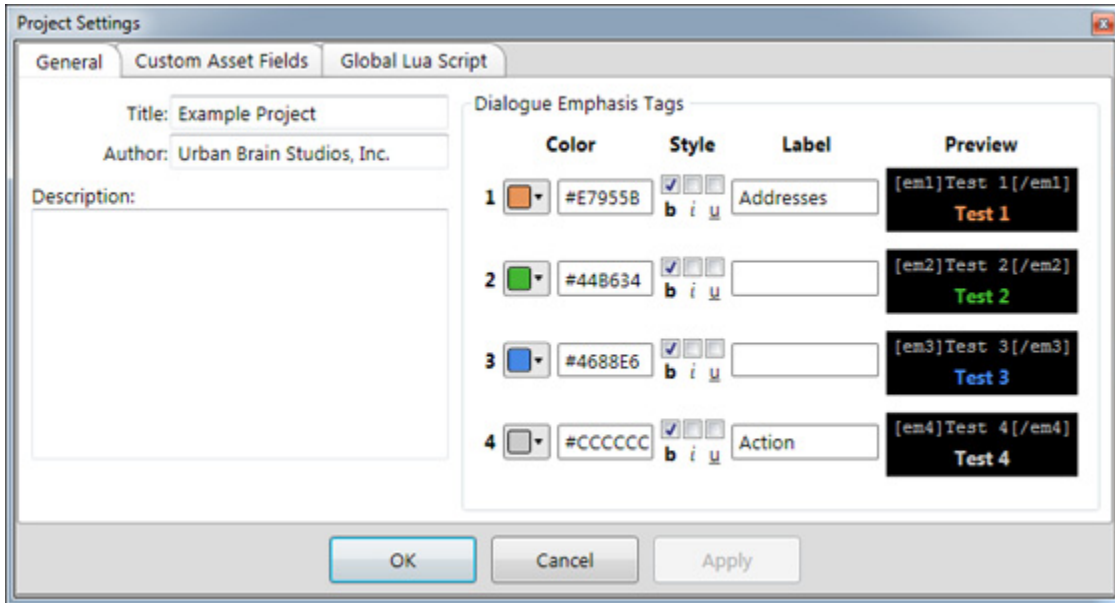
Adjusts the distance between adjacent levels of the tree.

Background Opacity

Sets the level of opacity for the conversation image that is displayed on the background of the tree diagram.

Project Preferences

Several other project-level preferences can be set by selecting the **Project | Project Settings...** menu option. Any settings here are saved with the project file and will be transferred to any computer opening the project.



Project Settings Window

The dialogue emphasis settings are used when simulating the conversation. There are four emphasis tags that you may use when creating your dialogue node which correspond to the four colors here. Additionally, labels can be assigned to each of the emphases and a key of labels will appear above the main workspace as a reminder.

Types of Assets

Assets are basically any noun in your project including Actors, Items, Locations, Conversations, and Variables. Even dialogue nodes are considered assets. Any asset fields that are defined will be available for editing in the properties window. Assets may also be sorted by asset field, or filtered by keyword.

Actors

A fundamental requirement for each dialogue node is to have an Actor and a Conversant. The Actor is the player or NPC who is currently speaking and Conversant is the player or NPC who is currently listening. Using the Asset Browser, any number of actors may be added to the project, each of whom can act as an actor or conversant for a given dialogue.

To add a new actor, either press the gear symbol on the actors asset group and select "New Actor", select the **Project | Add New | Actor** menu option, or use the **Ctrl + Alt + A** shortcut key. To delete an actor, you may select the actor from the actor list and then press the delete key, or again use the menu from the gear button.

Items and Locations

An optional component of a project is to define one or more Items and Locations. The Items and Locations can be used when defining conditions. Using the Asset Browser, any number of items may be added to the project in the same way that Actors were added.

Conversations

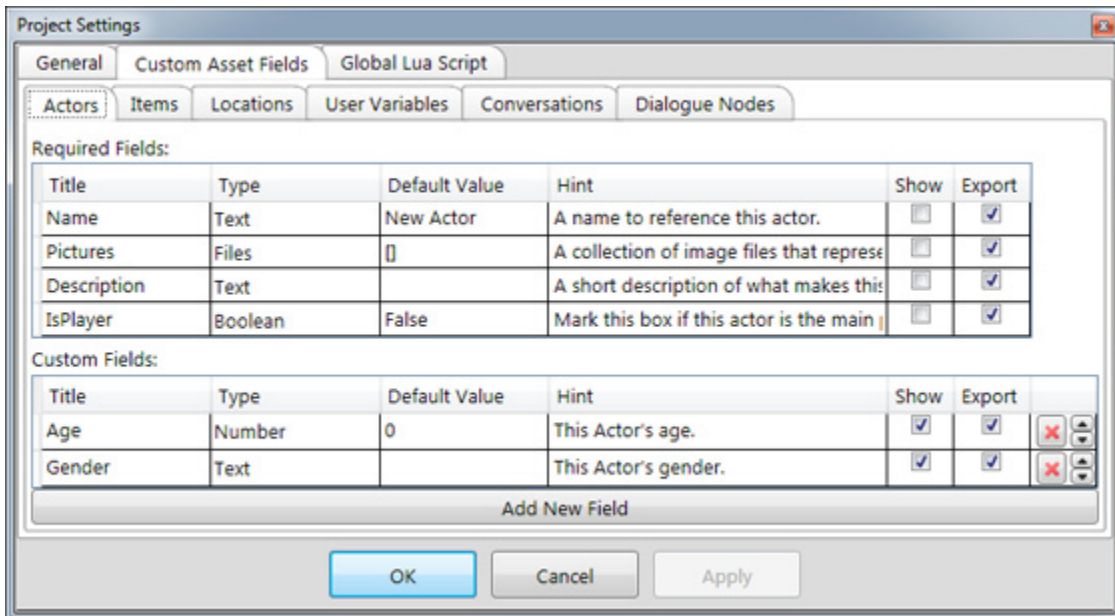
A fundamental requirement for each Chat Mapper project is to have one or more Conversations. The Conversations are used to hold dialogue nodes that make up the conversation. Each time a conversation is double-clicked on the list, a dialogue tree is opened in a new tab. Using the Asset Browser, any number of conversations may be added to the project. The conversations can be filtered or sorted just like any other asset.

Variables

Variables are another kind of asset that can be created to hold a piece of information to be used in conditions or scripts. A variable can also have a set of user-defined fields, but upon creation of the variable, a type must be assigned. The type can be **number** if the variable value is numeric, **text**, or **boolean** if the value will only ever be "true" or "false".

Asset Fields

Asset fields show up in the properties editor when clicking on any asset or dialogue node in the conversation tree. All assets have a set of pre-defined fields that are required. In addition, you may define even more fields to fit the need of your project. For example, it may be useful to have a location asset field called "HasVisited" that is a boolean and is set to true when the player visits the location. All asset fields can be accessed through Lua in conditions and scripts. Custom asset fields can be added using the setup screen after selecting the **Project | Project Settings... | Custom Asset Fields Tab** menu item.



Custom Asset Fields Setup

Possible types of asset fields are:

Number

Choose this if you will be adding, subtracting, or doing any other numeric operation.

Text

Just plain text that can be anything.

Boolean

The field will only ever be "true" or "false" and shows in the properties editor as a check box.

Files

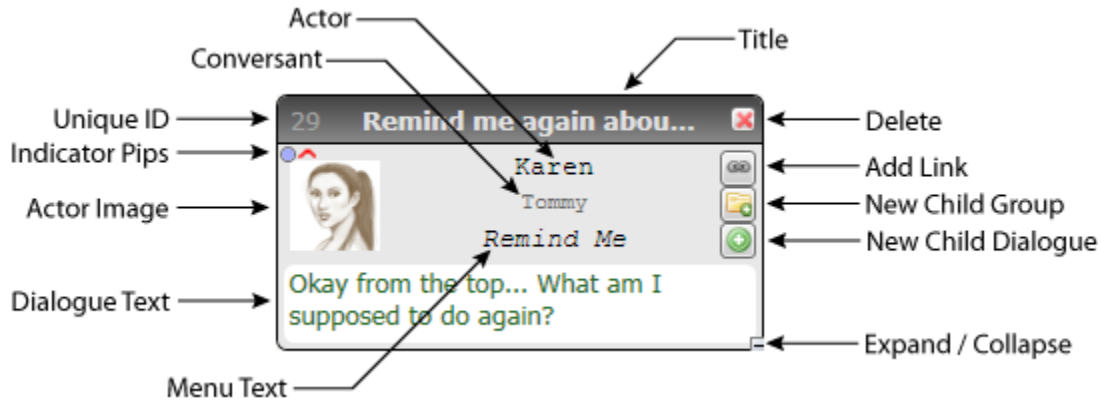
The field is meant to be a list of files chosen from the computer. In the properties editor this shows as a file list with options to add, modify, delete, and reorder the file list.

Actor / Item / Location

The field references an existing asset in the project. For the default value, you can specify either an asset ID number, or an asset name.

Anatomy

A dialogue node is displayed on the conversation tree as a small rounded rectangle showing the most important information in the dialogue as shown in the diagram below.



Unique ID

Each dialogue node in the entire project is assigned a unique ID to keep track of it in case two nodes have the same title.

Title

Displayed at the top of the node and in menus. This is not actually displayed during the simulation.

Actor / Actor Image

Shows the name and image of the actor who is speaking for this dialogue.

Conversant

Shows the name of the actor who is listening for this dialogue.

Menu Text

This is what is shown during the simulation as the menu option for the player.

Dialogue Text

The text that is displayed and spoken by the actor when this dialogue node plays.

Indicator Pips

Indication of whether there are conditions, scripts, or condition options assigned to this node.

Delete

Clicking this button will delete the dialogue node.

New Child / Group Dialogue

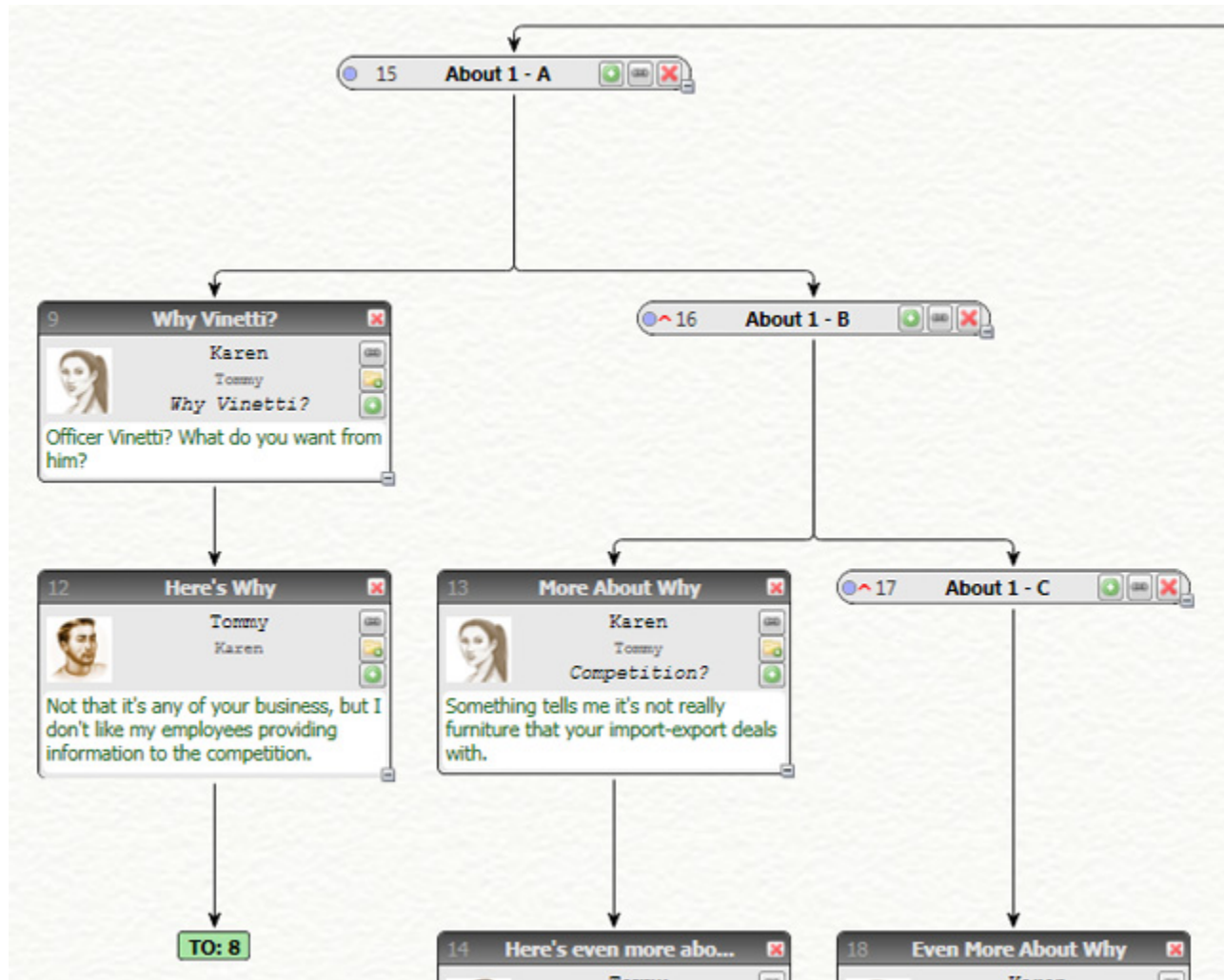
Clicking this button will create a new dialogue or group that is the child of the current dialogue. Holding control, shift, and control+shift has various other effects.

Make Link

Clicking this button will allow a link to be created to collect this dialogue node to another one. This is described in detail in the section on dialogue linking.

Group Nodes

Group nodes are used to create sub-trees within the overall conversation tree. This is useful from an organizational standpoint but also becomes very valuable in defining conditions as you will see in defining condition priorities. Groups can be thought of as simplified dialogue nodes and contain some of the same fields. Group nodes can also be added, arranged, and linked just like dialogue nodes.

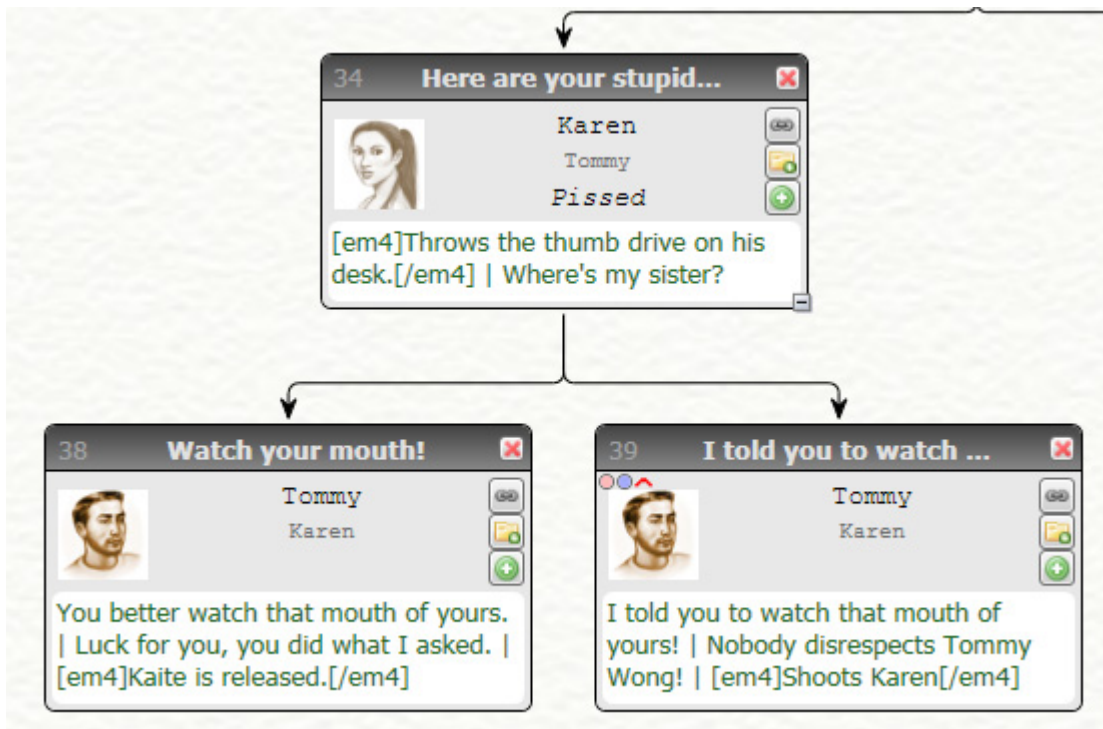


Group Nodes In A Conversation

Adding/Removing

When a conversation is added for the first time, the dialogue tree starts with a single Root Node from which all other dialogue nodes will link from. You may select the root node by left-clicking somewhere on the node that is not a button. To add a child node to the root node, or any other dialogue node, press the green plus button on the node that you wish to link from. Alternatively, you may use the context menu by selecting **Right-Click | New Child Dialogue**, or use the appropriate shortcut key with node selected. When first adding a dialogue node to the tree, you will be prompted to enter basic information about the dialogue node such as title, actor, conversant, and dialogue text.

Alternatively, you may add a sibling node to any node by selecting **Right-Click | New Sibling dialogue** or by holding shift while clicking the add button. The differences between a child node and sibling node are shown in the figure below. Nodes 38 and 39 are children of node 34, but they are siblings of each other.



To delete a dialogue node, select the dialogue node and then press the **Delete** key. Alternatively, you may press the red X at the top right corner of the dialogue node. The link will be reestablished when removing a node.

Linking

When you create a new node, it will automatically be linked in some way to a parent node in the conversation. Sometimes you might want to reorder a node within its sibling nodes, or add new links. One simple way to reorder nodes of the same level on the tree is to move the nodes left and right by selecting **Reorder Node Left** or **Reorder Node Right** from the node's right-click context menu. It is important to pay attention to the order of the nodes, because if there are multiple dialogue options that can possibly be displayed during simulation, then they will be presented to the player in the order that they appear on the tree from left to right (or top to bottom in the case of horizontal layout).

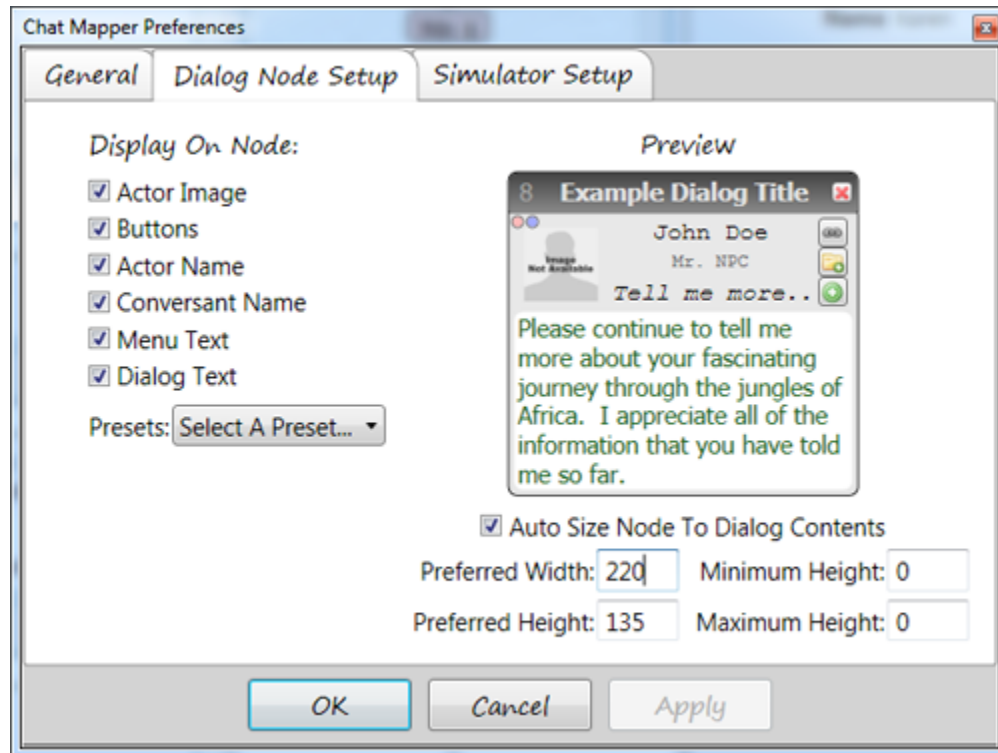
To delete a link, simply select it and press the **Delete** key.

To add a new link, first press the small button on the dialogue node showing the icon of a chain link. This will cause the conversation to enter into linking mode indicated by the red frame. Once you have entered linking mode, the next node that you click will be linked to the node that initiated the linking. For example, if you start linking from node 5, then click node 1, node 5 will instantly be linked to node 1. If you have the **Continuous Linking** option turned off in the preferences, then linking mode will continue until you press the **Escape** key.

If a link is made to a node that already has a parent, then a small connector node will appear to indicate the link and to simplify the complexity of tree display as shown below. You may click this connector node to edit the node that it links to, and you may also upgrade it to the primary link by selecting **Right-Click | Promote To Primary Link**.

Configuring

Several display preferences for the dialogue nodes can be set by selecting the **Tools | Preferences...** menu option and choosing the dialogue Node Setup tab as shown below.



Sections of the dialogue information that is shown on the node may be hidden by toggling the various check boxes, or a preset may be loaded by selecting from the drop-down list. The remaining options affect the size of the dialogue node:

Auto Size Node To Dialogue Contents

Each dialogue node will be set to the preferred width, but the height will be based on the length of the dialogue text.

Preferred Width

The width that you would like all dialogue nodes to be set.

Preferred Height

The height that you would like all dialogue nodes to be set unless they are auto sized to the dialogue contents.

Minimum Height

When auto-sizing the dialogue node, it will not be allowed to be smaller than this height.

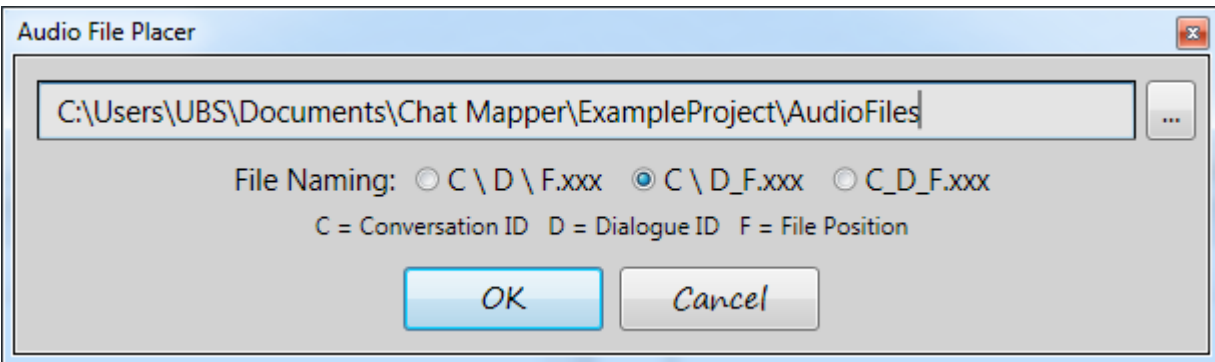
Maximum Height

When auto-sizing the dialogue node, it will not be allowed to be taller than this height.

Audio Files

Audio files in Chat Mapper are meant to represent dialogue voice-over and are played during simulation. Any number of files may be associated with a single dialogue node and each file is played as a segment on the dialogue as described in the section on Dialogue Markup. Chat Mapper utilizes any audio codecs installed for Windows, so if your file plays in Windows Media Player, it should play fine in Chat Mapper. To preview the audio for a dialogue node, select **Play Audio** from the right-click context menu of the node.

In order to make the job of assigning audio files to dialogue nodes easier, a tool is provided that can be accessed from the **Tools | Audio File Placer** menu item. This tool will allow you to automatically load in audio files based on the filename and directory structure of your audio folder.



Audio File Placer Tool

Available auto-detection options are:

Folder Organization

Assumes subdirectories exist for each Conversation ID, followed by subdirectories for each Dialogue Node ID, followed by files numbered for the position in the file list.

Example: `C:\003\002\005.mp3` would be loaded into conversation 3, dialogue 2, file position 5.

Folder/File Hybrid Organization

Assumes subdirectories exist for each Conversation ID, followed by files numbered with either DialogueID_FilePosition.xxx or DialogueID.xxx

Example1: `C:\003\002.ogg` would be loaded into conversation 3, dialogue 2, file position 1.

Example2: `C:\003\002_005.wav` would be loaded into conversation 3, dialogue 2, file position 5.

File Organization

Assumes all files are number as ConversationID_DialogueID.xxx or ConversationID_DialogueID_FilePosition.xxx

Example1: `C:\003_002.ogg` would be loaded into conversation 3, dialogue 2, file position 1.

Example2: `C:\003_002_005.wav` would be loaded into conversation 3, dialogue 2, file position 5.

Dialogue Markup

When writing dialogue text or menu text, there are optional tags that may be added to enhance and control the playback of the dialogue during simulation.

Menu Text Tags

[f] - Force Menu Display

Typically, if only one menu option is available to the player, it will automatically be chosen and played. This tag placed anywhere within the menu text will force the menu option to be displayed to the player and not auto-played.

[a] - Action Choice

This tag placed anywhere within the menu text simply causes the menu item to be displayed in italics.

Dialogue Text Tags

These tags can also be found by clicking the small pencil button above the dialogue text entry box in the properties editor.

| (pipe) - Break

During long dialogue text, it is useful to split the text into segments to be displayed in sequential order during playback. Simply placing the pipe character throughout the dialogue text will accomplish this. Each segment of the text will attempt to play the associated audio file position during playback as well.

Example: This is a very long dialogue text entry | that I would like to break up | into three different segments.

[pic=#] - Change Dialogue Picture

By default, the image of the actor of the dialogue node is displayed when the dialogue text is displayed during simulation. If any dialogue image files are defined, you may choose to display one of those pictures instead by inserting this tag with the # equal to the file position you wish to display, with the first file being position 1.

[pica=#] - Change Actor Picture

If more than one picture file is defined for the dialogue actor, you may display the file in the given position by using this tag.

[picc=#] - Change Conversant Picture

In rare cases, you may want to display a conversant's picture for the dialogue instead of the actor's. Using this tag, you may display the specified conversant picture file position.

[em#] some text [/em#] - Emphasize Text

You may specially emphasize any bit of dialogue text by surrounding the text with an [em] tag. The # should be replaced by a number 1 through 4 and the text will be colored given by the colors chosen in the project preferences.

About Lua

Lua is a scripting language widely used for games due to its simplicity and readability. Lua can be used to modify and store variables as well as call functions defined in the main application. Below is a brief primer on the Lua language but you can learn more by reading the [Lua 5.1 reference manual](#).

Variables

Variables in Lua are used to store a piece of information, such as a text value, numeric value, or a boolean value (true/false). A variable can be set using the equals sign:

```
variable_name1 = "text"
variable_name2 = value
variable_name3 = false
```

Note: The boolean variable is case-sensitive and must be lowercase.

One additional type of variable in Lua is called the Table. If you are familiar with programming, a table is much like a class where the table may store many additional variables, or properties, within it. A variable stored within a table can be accessed using either square brackets, or dot notation:

```
table_name["variable_name"] = value
table_name.variable_name = value
```

Arithmetic Operators

Lua supports all of the most common arithmetic operations on variables and values including addition (+), subtraction (-), multiplication (*), division (/), modulo (%), exponentiation (^), and negation (~). Note that Lua does not support the ++ or -- operators found in some programming languages, so to increment a variable value, you must use:

```
variable = variable + 3
```

Function Calls

If functions are available to the scripting language, they may be called by using the following notation:

```
function_name(parameter1, parameter2, ...)
```

Although some functions do not require any parameters at all. For more information on the functions available in Chat Mapper, see the next section.

Relational Operators

Relational operators always produce either a true or false response by comparing variables to some value or another variable. The most common operators are:

```
== : Equal - Compares two values or variables and returns true if they are equal.
~= : Not Equal - Returns the opposite value of the equal operator.
```

For a full list of operators, see the Lua reference manual.

Logical Operators

Logical operators combine multiple relational operators to allow for complex expressions. The available logical operators are `and`, `or`, and `not`.

.....
`20 == 20 or 20 == 10` returns **true** because the first half is true.

.....
`20 == 20 and 20 ==10` returns **false** because only one of the two statements is true.
.....

Using Lua in Chat Mapper

Chat Mapper supports the use of Lua 5.1 when writing scripts or conditionals. For both scripts and conditionals, all assets are defined as Lua tables with each asset field defined as a variable within the table. Each variable is named using the field title with spaces replaced with underscores.

Templates are provided for common scripts and conditions and can be accessed by clicking the pencil button above the script and conditions editors.

Note: All assets and dialogue nodes can be drag-and-dropped into the script or conditions window and will replace the selected text with the Lua asset table equivalent.

Special Asset Fields

In addition to the required and user-defined asset fields, two additional fields are available for use:

Status

Each asset gets a special status field that can be used as a general purpose tag to store some text value. This field does not have to be used, but is available for all assets.

Example: `Actor["Player"].Status = "dead"`

SimStatus

Each dialogue node asset has a special SimStatus field, used during simulations, which determines whether the dialogue has been offered as a choice to the player or has been displayed. Available values are "Untouched", "WasOffered", and "WasDisplayed".

Note: In some cases it is useful to wait to change the dialogue SimStatus to "WasDisplayed" until after the next menu options have already been built. In order to do this, use the "Delay SimStatus" option in the conditions editor.

Available Functions

Chat Mapper provides various functions that are accessible through Lua in order to perform special tasks beyond just setting and accessing variables.

Mutual Status

Chat Mapper can track a status that is defined between any two assets, which is referred to as a mutual status. To set or access a mutual status, use the following two functions:

```
SetStatus(Asset1, Asset2, "status value")
GetStatus(Asset1, Asset2)
```

Relationships

Chat Mapper will also track numerical relationship values between any two actors. Relationships can be useful for controlling NPC responses based on a relationship value. The following four functions can be used to control relationship values:

```
SetRelationship(Actor1, Actor2, "relationship type", value)
GetRelationship(Actor1, Actor2, "relationship type")
IncRelationship(Actor1, Actor2, "relationship type", incrementAmount)
DecRelationship(Actor1, Actor2, "relationship type", decrementAmount)
```

Variable Tracking

By Default, most variables are tracked automatically in the tracker panes of the simulation window. If, however you would like to add a variable to the tracker window manually, you may use the following functions:

```
TrackVariable(Table, "variable name")  
TrackStatus(Asset1, Asset2)  
TrackRelationship(Actor1, Actor2, "relationship type")
```

Conditions and Scripts

Conditions

The conversation just wouldn't work if all of the dialogue options were presented at all times. There must be a way to control which options get displayed depending on the player's inventory, the way the NPC feels about the player, etc. These are all controlled by defining Conditions.

Conditions are defined by clicking on a dialogue node or group node and typing a logical Lua expression into the conditions editor. All conditions are defined to allow the dialogue node to be offered or played if the condition evaluates to true. As such, you should ensure that the condition is written in such a way that it always evaluates to either **true** or **false** by using logical and relational operators.

Example:

```
Dialog[4].SimStatus ~= "WasDisplayed" and Variable["points"] > 15
```

Will allow the dialogue to be displayed if Dialog 4 has not been displayed before and the variable "points" is greater than 15.

Note: *You may include carriage returns to make the conditions look a little nicer and all lines will be reduced to a single line before evaluating the expression.*

False Condition Actions

By default if a dialogue node condition evaluates to false, the the node along with all following nodes will not be considered for inclusion in the following menu options. In some cases it is useful to allow a node to simple pass through to the following nodes if the condition is false. In order to do this, select the "PassThrough" option for false condition action.

Condition Priorities

If several nodes all evaluate their conditions to true, then the highest priority nodes will be displayed first. This is useful if you want a node to be displayed when some condition is satisfied and you want that node to be the only option available to the player. In that case, you would make sure the priority is higher than the other nodes.

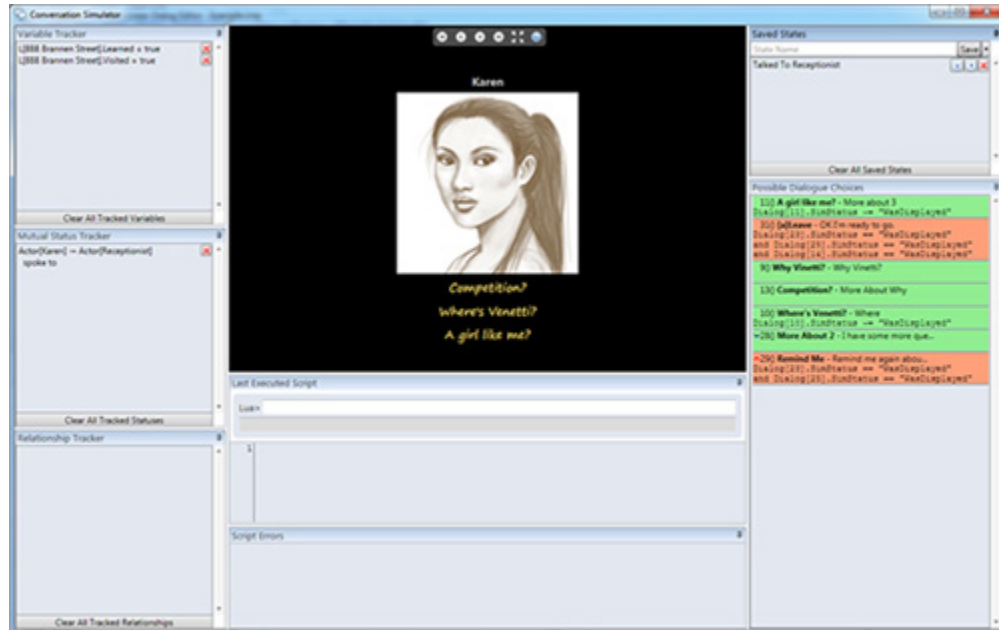
Note: *Priorities are compared only between nodes belonging to the same group.*

Scripts

Conditions would not be of much use if the variables and asset fields never change. In order to make changes to any asset field or variable, a Lua script is attached to a dialogue node and executed when the node is played. To see the Lua script attached to a node, simply select the node and look at the script editor. Each individual Lua command should be separated by a carriage return.

Simulation Interface

After you think you have the conversation exactly how you want, it is time to test it out in a mock-up of the game! From the menu, choose **Project | Simulate Conversation...** or press **F5** to start the simulator.



Conversation Simulator Window

Control Toolbar

At the top of the window are a set of controls for navigation and full-screen mode.

Tracker Windows

On the left side of the simulator, any time a variable, asset field, mutual status, or relationship changes, it will be added to the appropriate tracker. Tracked variables can be removed by clicking the button with the red X, and variables may be manually added by using the appropriate Lua function.

Lua Script Panel

At the bottom of the simulator window in a Lua command line, a text area showing the script that was run for the displayed dialogue, and a list of any Lua script errors. *Note: When using the command line to display a variable value, you must use the word "return" before your variable.*

Saved States

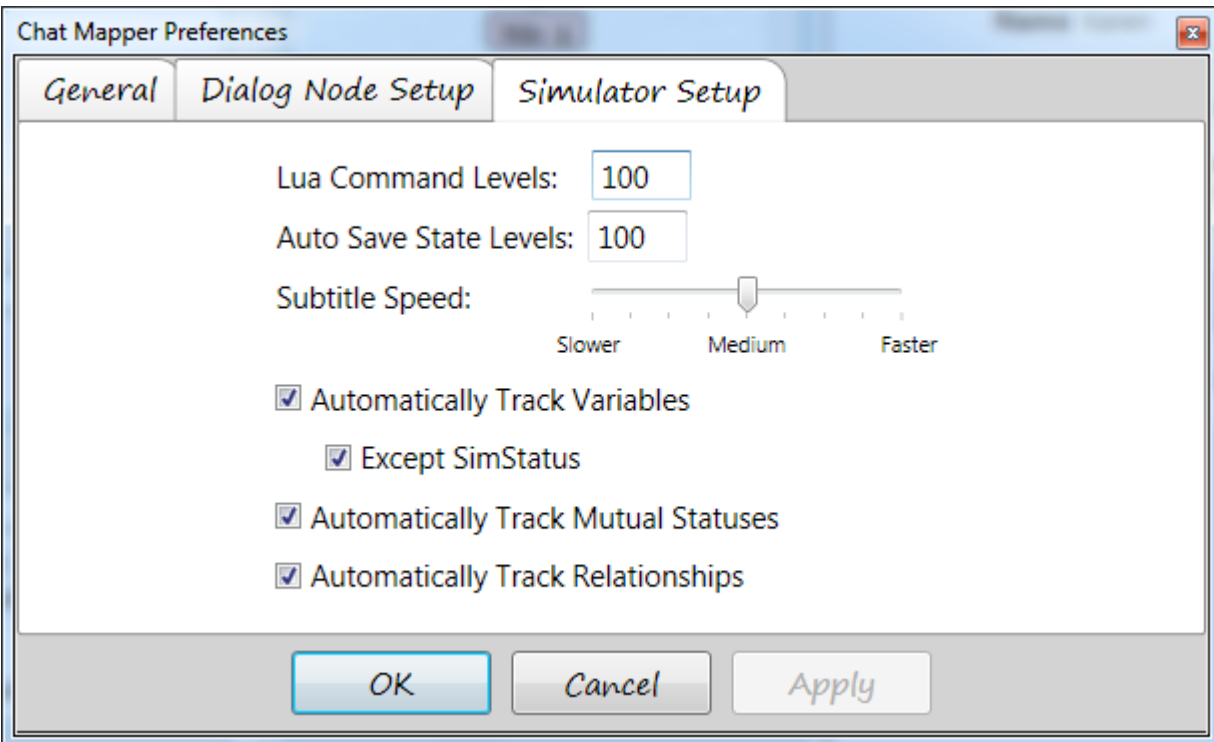
At the top right of the window is a list of all saved states. A state will store the values of all variables and asset fields as well as your current position within the conversation tree. To load only the variables from a state, click the button with the V and to load the entire state, click the blue play button.

Possible Dialogue Choices

At the bottom right of the window is a list of all possible dialogue choices that could be displayed to the player. Green items are dialogue nodes for which the conditions evaluated to true and red items evaluated to false. Also displayed are the condition priorities, the dialogue ID, and the group number that the dialogue nodes belong to. Remember that priorities are only compared against dialogues of the same group.

Simulator Preferences

Several options for the conversation simulation can be set by going to the **Tools | Preferences... | Simulator Setup Tab** menu item.



Simulator Setup Dialog

Lua Command Levels

Sets the maximum number of lua commands that are stored in memory from the command line. Pressing the up and down arrows keys allows you to scroll through the previously run command line entries.

Auto Save State Levels

Each time a menu is displayed that requires player input during simulation creates an automatic saved state in order to enable navigation. This value is the maximum number of those saves states that is put into memory.

Subtitle Speed

If no audio file is present for a dialog text, the on-screen time for the dialogue text is automatically calculated based on character count and punctuation. This setting allows for slower or faster subtitle speed.

Automatically Track Variables / Mutual Statuses / Relationships

If this is checked, any variables, statuses, or relationships that are changed will automatically be added to the associated tracker.

Except SimStatus

If this is checked, changes to the dialogue node SimStatus will not be tracked.

Basic Exporters

After you are satisfied that your conversation simulates properly, there are numerous export options to get your conversations into the hands of programmers, producers, or designers. The available exporters will be displayed when selecting the **File | Exporters** menu option. Commercial license holders may create their own exporters which will also be displayed in this list.

Package As CMPKG File: This export option is only available for commercial license holders and allows the author to export the Chat Mapper project and all associated media files to a compressed archive suitable for sharing. Another author may import the package file and view all of the media originally used.

Project As XML File: This is probably the most versatile export option because the resulting file contains all aspects of the project including assets, dialogue, and conditions in a human-readable and computer-readable plain-text XML format.

This option is only available to license holders.

Script as RTF File: Exports the script (as shown in the script view tab) to an RTF file, which includes formatting and font selections that can be opened and edited by word processors, such as Microsoft Word.

This option is only available to license holders.

Assets as CSV File: Exports the list of assets and all fields of each asset to a comma separated values (CSV) file that can be opened by spreadsheet software, such as Microsoft Excel.

Dialogue Graph as JPEG / PNG Image: Exports the entire dialogue tree as a JPEG or PNG image shown exactly as displayed in the software.

Technical Support

There are many ways to get help with Chat Mapper.

Documentation

You are on the right track already by reading through this documentation. Updates to the documentation will be posted to the [Chat Mapper website](#) as well.

Help Desk

If for some reason you cannot find an answer to your question, there is an official [help desk](#) where you can ask questions and view solutions to common problems.

Video Tutorials

If you need some more help getting started using all of the features, check out the [online video tutorials](#).

Index

A

- Assets, 3, 13
 - Actor, 13
 - Asset Fields, 14
 - Boolean, 14
 - Files, 14
 - Number, 14
 - Text, 14
 - Conversant, 13
 - Conversation, 13
 - Item, 3, 13
 - Location, 13
 - Variables, 13
- Audio Files, 20

C

- cmpkg, 6, 29
- Commercial Version, 4

D

- Dialogue Node, 15, 17, 18
 - Adding, 17
 - Group, 16
 - Linking, 18
 - Removing, 17
 - Root Node, 17

E

- Exporters, 20, 29

F

- Free Version, 4

G

- GUI, 7, 9, 10
 - Asset Browser, 7
 - Conditions Editor, 7
 - Dialog Tree, 7, 10
 - Error List, 7
 - File Browser, 7
 - Overview, 7
 - Properties Editor, 7
 - Script Editor, 7, 10

I

- Indie Version, 4

L

License, 4

N

Non-Linear Dialogue, 2

P

Package, 6, 29

Preferences, 11, 19

- Child Node Distance, 11

- Continuous Linking Mode, 11, 18

- Dialog Node Setup, 19

- Disable All Extensions, 11

- Disable Welcome Screen Fade Effect, 11

- Layout Direction, 11

- Show Welcome Screen, 6, 11

- Sibling Node Distance, 11

- Simulator Setup, 28

- Undo Levels, 11

Project Settings, 12

- Custom Asset Fields, 14

- Emphasis, 12

S

Scripting, 21

- Dialogue Markup, 21

- Lua, 22, 24

 - Conditions, 26

 - Functions, 24

 - Scripts, 26

 - Special Asset Fields, 24

- Tags, 21

Shortcut Keys

Simulator, 27

- Possible Dialogue Choices, 27

- States, 27

- Tracker, 27

Support, 30

System Requirements, 2

T

Template, 6

Tools, 20

- Audio File Placer, 20

W

Welcome Screen, 6

- Example Project, 6

- Learn Chat Mapper, 6

- New Project, 6

- Open Project, 6

- Recent Projects, 6

Show On Startup, 6

X

XML, 14, 29