JLC Dialogue Converter

This is a simple converter for the Dialogue System to import/export conversations to and from a text document. It supports dialogue branching, and retains node-specific sequence commands and conditions. Also included is a script for Google Docs to auto format exported text.

Example Conversation

#1 ExampleConversation

Ben: Hi Joe! {Camera(Wide, Scene, 1);}
Joe: Hi Ben.
Ben: What's your favourite colour? {Camera(Close, Joe, 1);}

[RED]
Joe: I like red.
Ben: Bold choice!
Ben: Red suggest vitality!
[RED]

[GREY]
Joe: I've always been partial to grey.
Ben: Grey?! Is that even a colour?
Joe: What can I say, I'm a grey kind of guy…
[GREY]

Ben: Well I can't stand around chatting all day.
// This is a comment that gets stored in the Description field of the preceding entry //

Ben: See you later!

[PARIS]
Joe: Actually. I wanted to ask you about your trip to Paris. |Variable["VisitedParis"] == true|
Ben: Oh it was great. Thanks for asking.
Joe: Great. See you later then.
[PARIS]

[GOODBYE]
Joe: Ok, bye. |Variable["VisitedParis"] == false|
[GOODBYE]

Syntax

#1 ConversationTitle => Marks the start of a new conversation. If you are overwriting an existing conversation in the database, the id must match.

ActorName: => Each new line creates a new node in the conversation. The actor name should match an actor in the database and must be followed by a colon. You can use abbreviated names by adding them to the ActorAbbreviations scriptable object in the DialogueConverter window.

{Sequence} => Sequences go directly after the line of dialogue or on a new line, and are wrapped in curly braces.

|Condition| => Conditions go after sequences and are wrapped in pipes. Conditions should follow sequences, if both are required. Can be placed on a new line.

±Script± => Scripts go after conditions and are wrapped in the "plus-minus" symbol. Scripts should follow conditions, if both are required. Can be placed on a new line.

[BRANCH] => Conversation branches are marked by brackets. The name of the branch is copied to the "Menu Text" field in the database. Each branch must have an end tag. You can nest branches. Response menu sequences can be included by adding a {Sequence} tag after the first [Branch] tag. Indentation is optional.

// COMMENTS // => Comments are stored in the Description field of the following dialogue entry. If placed before the conversation title, the comment will be stored in the Description field of the Conversation. Comments must begin and end with a double forward slash, but can span multiple lines.

Google Docs Script

The .gs file supplied can be used to auto format scripts in Google Docs. To enable and use the script:

1. Select Tools > Script Editor in your Google Doc
2. Copy/Paste the gs code into the script window.
3. Save the script.
4. A new menu item will appear called “Dialogue Editor”. Select “Format Text” from the dropdown.
5. The first time you run the script Google will warn you that the “app isn’t verified”. You will have to select “Advanced”, “Go to Dialogue Formatting”, “Allow” in order for the script to run.
6. Wait for the script to finish formatting the text before editing the document. This can take a while with very long scripts.