The Story File Format

Michael A. Gohde

February 22, 2017

1 Overview

This document exists to provide a series of tutorials and reference guides on how to write Nudge storylines using the Nudge storyline format.

1.1 How is a story file format useful?

Due to their open-ended nature, Nudge storylines can be extremely complex and difficult to represent. This is compounded further by the fact that the Nudge software stores each storyline in something on the order of five different tables in a databse. As such, it is excessively time consuming for humans to create, verify, and install their storylines by hand.

This file format aims to decrease the complexity of story authorship by enabling software to perform many of the repetitive or difficult tasks commonly encountered when writing a storyline.

1.2 What are some of the software tools available to story authors?

There is a fairly complete story validation and installation toolset available. Some of the functions performed by the toolset include:

1. Automatic story validation, including completeness and recursive reference checks.

2. Story graph generation (so that story authors may visualize their storylines).

3. Automatic story listing generation (so that story authors may see every possible combination of user choices).

4. Story installation and removal.

2 Example Storyline

This tutorial will begin with an example storyline and a description of every element within that storyline.

```
1  Title: Extremely Simple Story
2
3  Node_1:
4      This text will be presented to the user.
5
6      Responses:
7          Proceed -> 100% to End_Node
8
9  End_Node:
10     You have successfully completed a story.
11
12     Responses:
13         End -> 100% to END
```

2.1 Line by line summary

The above listing should be fairly straightforward. It presents a story in which there are two decision points, named "Node_1" and "End_Node". These decision points have text that is displayed to the user in addition to one possible answer choice with which the user can advance the story.

The story's title was declared in this line:

Title: Extremely Simple Story

This line consists of two distinct elements:

1. A noun followed by a colon.

2. Some text describing that noun.

As you will see, there are a number of nouns recognized by the story parsing software[1]. They each have some special meaning when checking, formatting, and displaying each storyline.

In this case, the noun is "Title", which instructs the parser to set the title equal to the value provided.

Node_1:

The above defines a new story "node", or individual decision point in the storyline. Note that it follows roughly the same structure as the title declaration shown above.

This text will be presented to the user.

This line (which is indented by one level[2]) defines the text that will be shown to the user when they reach the specified story node. Note that because this line is indented, it is used in describing the node that it is a part of.

Responses:

The above defines a block that will contain every possible answer choice that a user may have at the given node.

Proceed -> 100% to End_Node

Response definition lines follow a unique format that will be covered in more detail later. The general format is:

Answer choice text (presented to user) -> probability% to
Destination_Node_Name

The next story node follows the same conventions as the first, except for one important difference in its response line:

End -> 100% to END

"END" is a special node used by the format and story validation tools to define a choice that will immediately end the story.

[1]The Nudge Support Toolset actually uses a different, less human readable file format internally. The format described here was designed to be most easily readable by humans.

[2]Each "level" of indentation is a fixed number of spaces or tabs placed at the beginning of a line. Every line inside a block must have exactly the same indentation level as every other line in that block.

3 More about blocks

As you have seen, many of the statements shown above are followed by a series of indented lines. These lines make up the contents of a story block, and they can be thought of as attributes of whatever noun started the block.

All blocks follow the same format:

```
block_name:
(spaces or tabs) line 1
(spaces or tabs) line ...
```

It is possible to put any number of blocks inside of a block. For example:

```
block_name:
(spaces or tabs) another_block_name:
(more spaces or tabs) line ...
```

In the above example, block_name contains another block named another_block_name, which contains a line of text.

4 A more complex example

The following example makes use of every feature available to story authors using the story file format.

```
Title: Example story

D1_0:
    You are confronted with a serious question:
    To cheese it or not to cheese it?

    Responses:
        Cheese it! -> 50% to D2_0, 50% to D2_1
        Don't cheese it! -> 80% to D2_2, 20% to D2_3

D2_0:
    Note: "Note" and "Comment" are special nouns in the
        file format. They direct the parser to ignore
       anything that comes after them on their line.
    You were able to cheese it!
```

```
14
15      Responses:
16          Proceed -> 100% to D3_0
17
18  D2_1:
19      Comment: This is also a comment.
20      You were unsuccessful at cheesing it!
21
22      Responses:
23          Proceed -> 100% to D3_0
24
25  D2_2:
26      Comment: Each story node may contain several text
            or blank lines as long as they all share the
            same indentation level.
27      You proceed not to cheese it.
28      This is an additional line to test the parser.
29
30      Yet another line!
31
32      Responses:
33          Proceed -> 100% to D3_0
34
35  D2_3:
36      You panic and cheese it anyway!
37
38      Responses:
39          Proceed -> 100% to D3_0
40
41  D3_0:
42      Regardless of whether you cheesed it, something
            happened.
43
44      Responses:
45          End -> 100% to END with (Examplereward; amazing
              description; 27)
```

Despite a seemingly much greater complexity, only three additional features are demonstrated in this story:

1. It is possible to leave notes in any story node.
2. Multiple destinations can be provided for each answer choice. To take advantage of this, simply put a comma between every destination definitionn.
3. Users can be given badges on END nodes. The syntax for the statement that does this is: (Reward name; Description of the reward; Number of points awarded)

5 Specific rules and quirks

Since this file format is intended to be machine-readable, several strict rules need to be followed in order to produce correct output in all cases[3]:

1. Lines generally should not have any unused spaces or tabs at the end.
2. Try to use either spaces or tabs for indentation. Mixing spaces and tabs may produce undesired and unpredictable results.
3. Every block must have consistent indentation throughout. If a line is given greater or lesser indentaiton than it should have, then the parser may drop it.
4. The spacing in response lines may not be strictly necessary. For compatibility purposes, however, all stories should use the same spacing as the examples shown here.
5. It is unfortunately not possible to explicitly define paragraphs in a story node. Every line is merged after being stripped of leading spaces or tabs.
6. You may include as many comments in any given story node as you would like.
7. An individual "Comment:" or "Note" statement will only cause the parser to discard the remainder of the line it is on.

8. While the format allows for theoretically infinitely many responses per story node, Nudge cannot handle more than 100.

9. There may be some slight variance between the weights as specified and how Nudge will interpret them.

10. It is technically possible for non-END destinations to have reward definitions. This behavior is, however, not implemented in Nudge at the moment.

11. All story node names should start with at lest one letter.

12. All story title definitions should be on teh first line of the story.

[3]It is possible for a story containing some of the problems listed in this section to compile and install correctly. However, this behavior may change with future revisions to the story parsing software.