novelWriter

Project File Format 1.5 Specification

This document covers the file format specification for the 1.5 file format for novelWriter project files. See the documentation[1] for a full description of the app's functionality. The file format was intrioduced in Release 2.0.

The target audience of this document is developers who intend to write a tool or script that generates valid novelWriter project files from data from other applications, or for templating purposes.

Note that flags are generally written out as yes/no values, but also true/false and on/off are understood by the application. The None value that occasionally occurs in actual project files is the text representation of the Python None value, and indicates that there is no value set.

Revisions

1. From Release 2.1 Beta 1: Removed the titleFormat node from settings.
2. From Release 2.3 Beta 1: Removed the title node from project.
3. From Release 2.3: Added TEMPLATE item class.
4. From Release 2.5: Added the shape node

XML Root

Tag: novelWriterXML

The root tag of a novelWriter project file must be named "novelWriterXML". This value is case sensitive. The fileVersion attribute must also be set to the appropriate format version in order to be parsed correctly.

Attributes

Name	Required	Description
appVersion	No	A string representation of the novelWriter version used to write the file. The value is not required, but is used to report to the user when the file format is converted.
hexVersion	No	A hex representation of the novelWriter version used to write the file. The value is used to check if the project is opened by a lower version than was used to write it, which issues a warning. Defaults to 0x0.
fileVersion	Yes	The file format version used when writing the project file. This determines how the file is parsed, so it is important that it is set correctly. Valid values are "1.0", "1.1", "1.2", "1.3", "1.4", and "1.5".
fileRevision	No	The revision of the file format used for non-breaking changes. This is an interger value.
timeStamp	No	The ISO 8601 timestamp of when the file was saved. This information is not used by the parser, and is purely for debugging.

Example

```
<novelWriterXML appVersion="2.0-rc1" hexVersion="0x020000c1"
                fileVersion="1.5" fileRevision="1"
                timeStamp="2022-11-05 17:46:51">
```

1 See: https://docs.novelwriter.io

Project Node

Tag: project

The project section is a level one node under the root. It stores the primary settings for the project. The only required values are the id attribute and the name value. All other entries are set to default values during parsing if ommitted from the XML.

Attributes

Name	Required	Description
id	Yes	A UUID that is unique for the project and does not change during the lifetime of the project. It is used by novelWriter to keep track of temporary files and to associate cross-project settings with specific projects without relying on the project name.
saveCount	No	A number representing the number of times the project file has been written as a result of user interaction with the application.
autoCount	No	A number representing the number of times the project file has been written as a result of internal timed automatic operations of the application.
editTime	No	A number representing the accumulated time in seconds the project has been open.

Values

Name	Required	Description
name	Yes	The name the user has given to the project.
title	No	Removed in Revision 2.
author	No	The name of the novel author or authors. The value is used in some places in the user interface, and can be added to manuscript builds.

Example

```
<project id="e2be99af-f9bf-4403-857a-c3d1ac25abea"
         saveCount="5" autoCount="10" editTime="1000">
    <name>Sample Project</name>
    <author>Jane Smith</author>
</project>
```

Settings Node

Tag: settings

The settings section is a level one node under the root. It stores the various runtime values for the project, and the settings available from the Project Settings panel. None of these settings are required, and if missing, will be set to default values. The entire settings section can thus be ommitted.

Attributes

None

Values

Name	Required	Description
doBackup	No	User-controlled setting for whether or not to run backup when the project is closed.
language	No	The language used when building the manuscript. The setting is controlled from the Build Novel Project tool.
spellChecking	No	The spell checking language to use for this project if it differs from the default spell checking language for the application. Attribute auto: A flag determining whether the spell checking is automatic or not.
lastHandle	No	An auto-generated list of the last documents open in the editor and viewer, and which novel root folder was last viewed in the Novel Tree and Outline View. These are saved as key/value pairs. See separate section for how they are stored as XML.
autoReplace	No	The entries of the auto-replace feature available from Project Settings. These are saved as key/value pairs. See separate section for how they are stored as XML.
titleFormat	No	Removed in Revision 1.
status	No	The status labels as defined by the user in Project Settings. These are saved as special key/value pars. See separate section for how they are stored as XML.
importance	No	The importance labels as defined by the user in Project Settings. These are saved as special key/value pars. See separate section for how they are stored as XML.

Example

```
<settings>
    <doBackup>yes</doBackup>
    <language>en_GB</language>
    <spellChecking auto="yes">en_GB</spellChecking>
    <lastHandle />
    <autoReplace />
    <status />
    <importance />
</settings>
```

Key/Value Nodes

The key/value nodes are used for lastHandle, autoReplace and titleFormat settings. The lookup key is stored as an attribute, and the value is the text of the node.

Attributes

Name	Required	Description
key	Yes	The lookup key.

Example

```
<lastHandle>
    <entry key="editor">636b6aa9b697b</entry>
    <entry key="viewer">636b6aa9b697b</entry>
    <entry key="novelTree">7031beac91f75</entry>
    <entry key="outline">7031beac91f75</entry>
</lastHandle>
```

Status/Importance Key/Value Nodes

The status and importance settings are stored as key/value nodes with additional data attributes. The label of the status or importance setting is saved as the node text value.

Attributes

Name	Required	Description
key	Yes	The lookup key. For status labels, it must consist of an “s” followed by six hexadecimal numbers. For importance labels it must consist of an “i” followed by six hexadecimal numbers. Each key string must be unique. Internally, they are random numbers, but also sequential values are valid.
count	No	A record of the number of times each label is used in the project content.
red	Yes	A number between 0 and 255 representing the red component of the label colour.
green	Yes	A number between 0 and 255 representing the green component of the label colour.
blue	Yes	A number between 0 and 255 representing the blue component of the label colour.
shape	No	An icon shape. Allowed vaues are SQUARE, TRIANGLE, NABLA, DIAMOND, PENTAGON, HEXAGON, STAR, PACMAN, CIRCLE_Q, CIRCLE_H, CIRCLE_T, CIRCLE, BARS_1, BARS_2, BARS_3, BARS_4, BLOCK_1, BLOCK_2, BLOCK_3, BLOCK_4. Defaults to SQUARE. Added in Revesion 4.

Example

```
<status>
    <entry key="sf12341" count="4" red="100" green="100" blue="100"
           shape="SQUARE">New</entry>
    <entry key="sd51c5b" count="0" red="193" green="129" blue="0"
           shape="SQUARE">Draft</entry>
    <entry key="s78ea90" count="1" red="58" green="180" blue="58"
           shape="SQUARE">Finished</entry>
</status>
<importance>
    <entry key="ia857f0" count="5" red="100" green="100" blue="100"
           shape="CIRCLE">None</entry>
    <entry key="icfb3a5" count="2" red="0" green="122" blue="188"
           shape="CIRCLE">Minor</entry>
    <entry key="i2d7a54" count="2" red="21" green="0" blue="180"
           shape="CIRCLE">Major</entry>
</importance>
```

Content Node

Tag: content

The content section is a level one node under the root. It stores all the project items of the project. The items are stored in the order in which they appear in the project tree, so altering the order will affect the project structure. The attribute with the order number is not used when processing the data. It is written to the XML for debugging purposes.

Attributes

Name	Required	Description
items	No	The number of project items in the content section.
novelWords	No	The number of words in total for all document nodes with layout DOCUMENT. The value defaults to 0, but this number forms the basis of computing writing statistics during a session, so the number should be properly set to ensure correct statistics.
notesWords	No	The number of words in total for all document nodes with layout NOTE. The value defaults to 0, but this number forms the basis of computing writing statistics during a session, so the number should be properly set to ensure correct statistics.

Values

Name	Required	Description
item	No	A node representing a project item.

Example

```
<content items="27" novelWords="954" notesWords="409">
    <item handle="7031beac91f75" parent="None" root="7031beac91f75"
          order="0" type="ROOT" class="NOVEL">
      <meta expanded="yes"/>
      <name status="sc24b8f" import="ia857f0">Novel</name>
    </item>
    ...
</content>
```

Item Nodes

The item nodes make up the actual project content of the project. Each node represents either a root folder, a regular folder, or a document. Each node has a type, class and layout setting that determine its category. Each item is given a handle that is a random hexadecimal string of length 13.

For the items that are document files, the handle corresponds to its expected filename. It is therefore important that they match in generated projects. Each document node is expected to correspond to a file in the contents folder of the project named “`content/7031beac91f75.nwd`” for the item handle “`7031beac91f75`”. If there is no such file, the document item is assumed to contain no text, and a file is created when the user tries to save text to it.

Each item node has a meta data node and a name node. The meta data node contains only collected information, and is thus not strictly required. However, an accurate word count and setting the correct icon in the project tree depend on these values. These values are set by the indexer class, so rebuilding the index should restore the data.

The name node contains the primary user defined settings for an item, and is required for the entry to be valid.

Attributes

Name	Required	Description
handle	Yes	The 13 value hexadecimal string that represents the item in the project. This is the primary identifier of a project item, and is required. It must be unique withing a project. Internally, it is generated as a random value, but it can also be sequential.
parent	Yes	The handle of the parent item in the tree. This value should only be None for root folders. All other items must have a parent handle set. If the parent handle is None for an item that isn't a root folder, the item will be treated as orphaned during project loading.
root	No	The handle of the top of its hierarchy of parent items. That is, the root folder which it ultimately sits under. If the attribute is not set, it will be computed during loading. It is saved to the XML file for efficiency reasons.
order	No	The numerical order of the item under its parent item. This value is not used during the loading process as the physical item order in the content node is used instead. It is primarily saved to the XML for debugging purposes.
type	Yes	The item type of the item node. Allowed values are ROOT, FOLDER and FILE.
class	Yes	The item class of the item node. There are a number of item classes available in the app, all corresponding to a specific type of root folder. Each item in a root folder should have the same class set as the root folder itself. The attribute is not strictly required for items that aren't root folders as it will be automatically set to match during loading. Allowed values are NOVEL, PLOT, CHARACTER, WORLD, TIMELINE, OBJECT, ENTITY, CUSTOM, ARCHIVE and TRASH. Added in Revision 3: TEMPLATE
layout	Yes	The item layout of the item node. Allowed values are DOCUMENT and NOTE. This is the attribute that determines if a FILE type item is a Novel Document or a Project Note.

Values

Name	Required	Description
meta	No	A node of meta data attributes for the item.
name	Yes	A node of user settings for the item.

Meta Nodes

The meta data collected for the current item. This data either represents the item's last state in the app, or consists of data collected by the indexer. It can be restored by rebuilding the index, so the data is not essential.

Attributes

Name	Required	Description
expanded	No	Whether the tree node in the project tree was expanded or collapsed during the last session. Applies to all item types, but is ignored for items without child items.
heading	No	The heading level of the first heading of the text of the item. Only applies to FILE item types. Allowed values are H0, H1, H2, H3, and H4. Other values are reset to H0.
charCount	No	The number of characters in the text of the item. Only applies to FILE item types.
wordCount	No	The number of words in the text of the item. Only applies to FILE item types.
paraCount	No	The number of paragraphs in the text of the item. Only applies to FILE item types.
cursorPos	No	The last cursor position in the text of the item from the last editing session. Only applies to FILE item types. The value is used to restore the cursor position when the document is opened in the editor.

Name Nodes

The name node contains information about an item that is set by the user. Including its label, which is the text value of the node. This is the label that is displayed in the project tree.

Attributes

Name	Required	Description
status	No	The ID of the status label that has been set for this item. Defaults to the first status item defined in the status section of the settings node.
import	No	The ID of the importance label that has been set for this item. Defaults to the first importance item defined in the importance section of the settings node.
active	No	The active/inactive status of the document. Only applies to FILE item types.

Example

```
<item handle="53b69b83cdafc" parent="7031beac91f75" root="7031beac91f75"
      order="0" type="FILE" class="NOVEL" layout="DOCUMENT">
    <meta expanded="no" heading="H1" charCount="93" wordCount="19"
          paraCount="2" cursorPos="119"/>
    <name status="sc24b8f" import="ia857f0" active="yes">Title Page</name>
</item>
```