RSMFValidatorSampleCode

Introduction

The RSMFValidatorSampleCode project contains both a reference implementation of the RSMF Validator library and a precompiled GUI sample application that can validate RSMF files. This document also has several exercises that a user can run with the prebuilt GUI sample application to better understand how the Validation library works and how to solve the issues that the library might report.

Included with the sample code in the RSMFManifestSchema folder is rsmf_schema_2_0_0.json. This file describes the schema that is used to validate the data in rsmf_manifest.json files. It is provided to assist developers with creating their own applications that generate RSMF files. All rsmf_manifest.json files with a version of 2.0.0 must adhere to this schema.

Sample 1

This sample contains a rsmf_manifest.json file that contains content that violates the RSMF JSON schema. An application like notepad or notepad++ will be required to fix the faulty file.

1. Start ValidatorGuiCore.exe located in bin\ValidatorGui
2. Click the 'Select Files" button
3. Change the file type to 'json' in the lower right corner of the window that pops open.
4. Navigate to Documentation\Sample1 and open the rsmf_manifest.json file located there.
5. After a few moments the sample application reports that the file has failed validation.
6. Inspect the resultant information given in the application's window.
7. In this file the specific error encountered is listed. In the case of sample 1, there is an invalid participant. The participant is reported as being invalid because it's missing a required property.
8. Open rsmf_manifest.json in a text editor and locate the participant in rsmf_manifest.json. Hint: The line and position information are given as part of the error message.
    a. To correct the invalid participant, add an ID property with a valid string to the participant that is defined on line 4. Save the file.

```
{
"id": "1",
"display":"Participant 1"
}
```

9. After the rsmf_manifest.json file has been corrected reopen the file following from step 2. The file is now reported as passing.

Sample 2

This sample contains an rsmf.zip file that passes with warnings.

1. Start ValidatorGuiCore.exe located in bin\ValidatorGui
2. Click the 'Select Files" button
3. Change the file type to 'zip' in the lower right corner of the window that pops open.
4. Navigate to Documentation\Sample2 and open the rsmf.zip file located there.

5. The application reports that the file rsmf.zip has passed with warnings. In this file there is a participant that references an avatar, but that avatar is not included in the Zip file.
6. Warnings are errors that may not cause issues in Relativity. While this warning may not cause issues in Relativity, for this exercise it is expected that all avatars will be present.
7. There are two ways to address this warning. The first is to remove the reference to the avatar in the rsmf_manifest.json file that is in the rsmf.zip file. As this avatar is expected to be present this isn't the best solution. Instead the solution is to add the correct avatar to the rsmf.zip file.
8. A file, avatar.png has been included in the Sample2 directory. The warning in the log file indicated that the missing avatar was named AV1. For the issue to be corrected, rename avatar.png to AV1. Notice that no extension is used in this example.
9. Add the AV1 file to rsmf.zip. In Windows this is as simple as selecting AV1 and dragging it on top of rsmf.zip
10. Rerun the sample application starting from step 2. The file rsmf.zip now passes.

Sample 3

This sample demonstrates how to address issues that occur at the RSMF EML layer. It is handy to have some knowledge of the MIME format (RFC 5322) but it is not necessary to follow the exercise. All RSMF files are just EML files with specific requirements and a '.rsmf' extension. See the RSMF specification on Relativity's documentation site for more information.

1. Start ValidatorGuiCore.exe located in bin\ValidatorGui
2. Click the 'Select Files" button
3. Open sample.rsmf located in Documentation\Sample3
4. The sample application reports that the file sample.rsmf has failed. There is an extra attachment at the RSMF EML layer.
5. Examine sample.rsmf in a text editor. Notice at line 17 a new section begins (this is designated by the --RSMFEML indicator). The headers for this section indicate that it is an attachment.
6. RSMF files are only allowed to have one rsmf.zip as an attachment. It is necessary to remove the second attachment from the RSMF file. Note that this isn't the same thing as removing an attachment from the rsmf.zip file.
7. To remove the attachment, delete all lines of the file from the --RSMFEML on line 17 up to but not including the next line that begins with --RSMFEML.
8. Rerun the sample application. The file sample.rsmf now passes.

Sample 4

This sample demonstrates how to address issues that take multiple passes of validation. It uses a rsmf_manifest.json file with several errors and warnings to demonstrate that some issues cannot be detected until previous issues are addressed.

1. Start ValidatorGuiCore.exe located in bin\ValidatorGui
2. Click the 'Select Files" button
3. Change the file type to 'json' in the lower right corner of the window that pops open.
4. Open rsmf_manifest.json located in Documentation\Sample4
5. The sample application reports that the file rsmf_manifest.json has failed. The report indicates there was a ManifestSchemaViolation. The property "id" expects a string value, not an integer.

6. Correct the issue by adding quotes around the id value in rsmf_manifest.json. Rerun the validation sample application starting at step 2.
7. Three new warnings are reported by the application. Warnings won't prevent the RSMF file from viewing, but it may not view as expected if the warnings are not addressed. The first two warnings that will be addressed are ManifestInvalidConversationParticipant and ManifestInvalidEventParticipant. The final warning, ManifestUndefinedProperty, shows that a property "attachment" isn't a property that is defined by the RSMF specification.
8. ManifestInvalidConversationParticipant and ManifestInvalidEventParticipant warnings are generated because the conversation lists participant "2" as a member, but there is no participant with an id of "2". It must be determined if there is a missing participant in the participants array or if the id in the conversation's participants array is incorrect. For this exercise it is assumed that the conversation's participants array is incorrect. To correct these issues, change the "2" to a "1" in the conversation's participants array.
9. The undefined property can be addressed in two ways. It's possible that the creator of this rsmf_manifest.json meant to include an attachment. If this were the case the correct property to use is the "attachments" property with an array of attachment objects. It might also be possible that the creator did intend to use their own property. In this case, the correct way to do this is to use the "custom" property with an array of name/value pair objects. For this exercise the assumption is that an attachment was meant to be included. Modify the rsmf_manifest.json so that the 'attachment' property becomes a valid 'attachments' property.
10. After the final issue has been addressed, rerun the validator sample application starting from step 2. The application reports that the file is now valid.