Document Conventions for Handling Long Lines in Artwork Containing Code

draft-wu-netmod-yang-xml-doc-conventions-00

IETF 101, London, UK

Qin Wu (bill.wu@huawei.com)
Adrian Farrel (adrian@olddog.co.uk)
Benoit Claise (bclaise@cisco.com)

Why this draft?

- Goal:
  - Documentation conventions that allow
    - YANG examples to be presented in IETF documentation when leaf node encoding would otherwise exceed the maximum line length.
  - No change to the rules for presenting YANG models or for encoding YANG in data files or on the wire.
- Motivation:
  - Many documents that define YANG modules also include examples presented in XML.
  - IETF documentation has specific limits on line length and some XML examples or json examples have to include line wraps that would not normally be allowed according to the XML representation rules of RFC7950 and RFC7952.
  - As stated in section 3.12 of draft-ietf-netmod-rfc6087bis-20 (Module Usage Examples), examples became extremely important in drafts. They need to validated via automation.
  - Before doing so, allow the presentation of such examples in a way that is easily parsed by a human reader
  - Not representative of how the XML must be presented to a software component or carried on the wire.

What does the draft do?

- Aims at Documenting Conventions for Expressing YANG in XML
- Target a common approach that can be used in all future YANG documents
  - Avoid each document inventing its own mechanism
- How:
  - Define convention for Splitting an Example Leaf Node Value Across Lines
  - Define convention for Representing XML Encodings of Metadata Annotations
  - Define rules for parsing Valid XML From Examples
  - Provide boilerplate text to use when these conventions are used
    - So readers know how to interpret the examples

Splitting an Example Leaf Node Value Across Lines

```
Example a:Leaf Node With a Long String Value
   <long-leaf-string-node-label>
     Once upon a time, in a land far away, there lived a Great King.
   </long-leaf-string-node-label>
```

Equivalent to

```
Example b: Split A Long String Leaf Node value Across Lines
        <long-leaf-string-node-label>
          Once upon a time, \
            in a land far away, \
            there lived a Great King.
        </long-leaf-string-node-label>
```

Equivalent to

```
Example c: A Split Leaf Node Example
        <long-leaf-node-label>
          long-leaf-node-value
        </long-leaf-node-label>
```

Documentation Conventions:

- The broken line MUST be terminated with a backslash ("\") without the addition of any additional space before the backslash and with no further characters after the backslash.
- Any continuation lines MUST be indented with a whitespace offset of at least two characters.
- The document presenting the example MUST include the following statement:
  - The examples in this document adopt the conventions shown in BCP XX [RFCYYYY] for splitting node labels and node values onto separate lines. This convention is used to make the examples easier to read but does not change the encoding rules for the XML representation of YANG as described in [RFC7950].

Splitting an Example Leaf Node Value Across Lines

```
Example d:An Example Leaf Node With a Complex String Value
   <long-leaf-complex-string-node-label>
     Punctuation is important. As are line feeds.
     Some characters are special. E.g., the backslash \. Don't forget.
   </long-leaf-string-node-label>

Equivalent to:

Example e:An Example Leaf Node With a Complex String Value Split

        <long-leaf-complex-string-node-label>
          Punctuation is important. \
            As are line feeds.
          Some characters are special. \
            E.g., the backslash \. \
            Don't forget.
        </long-leaf-string-node-label>
```

- Document Convention:
  - When a backslash appears in the node value, the example MUST be arranged so that the backslash is not the final character of a broken line

Representing XML Encodings of Metadata Annotations

```
Example f: An Example Leaf Node With Metadata Split Across Lines
   <error-path
     xmlns:t="http://example.com/schema/1.2/config/verylongpathname\
       thatcannotfitononeline">
     /t:top/t:interface[t:name="Ethernet0/0"]/t:mtu/t:anotherattribute
     /t:afinalattribute
   </error-path>
```

- Document conventions:
  a. When an example XML representation of a leaf node element that includes metadata attributes results in a line being longer than the maximum number of characters allowed in a line of an IETF document, the value of the leaf node must be split across more than one line.
  b. Where possible, all line breaks should be inserted between metadata attributes.
  c. Continuation lines MUST start with a whitespace offset of at least two characters. The leading and trailing whitespace of each line MUST be ignored.
  d. The document presenting the example MUST include the following statement:
     - The examples in this document adopt the conventions shown in BCP XX [RFCYYYY] for splitting metadata annotation across multiple lines. This convention is used to make the examples easier to read but does not change the encoding rules for the XML representation of YANG metadata annotations as described in [RFC7952].

Rule for parsing Valid XML From Examples

- When parsing a leaf or leaf-list node in an example, the following rules should be applied to generate valid XML:
  - If a white space, carriage return, or line feed character is encountered between close (">") and open ("<") angle brackets it should be stripped.

  - If a white space, carriage return, or line feed character is encountered within a string value of a leaf node or leaf-list node, it should be preserved exactly as shown except in the following special case:
    - If a backslash character ("\") appears within the string value of a leaf node or leaf-list node and if and only if it is immediately followed by a carriage return or line feed character then all carriage return, line feed, and whitespace characters should be stripped until the next character is encountered.

  - If a white space, carriage return, or line feed character is encountered within metadata annotations, but not within quotes, it should be stripped. Parsing may expect the next valid character found to indicate the start of a new metadata attribute.

  - If a backslash character ("\") appears within the quoted value of a metadata attribute and if and only if it is immediately followed by a carriage return or line feed character then all carriage return, line feed, and whitespace characters should be stripped until the next character is encountered.

Next Step

- This is the not the biggest problem in the world, but we have to solve it.
- Next Step:

  a. Add JSON example

  b. Example validation automation via tooling