YANG Pretty Printer Introduction

Johannes Specht

Johannes Specht
Dipl.-Inform. (FH)

Kurfürstenwall 2
45657 Recklinghausen
North Rhine-Westphalia
GERMANY
M +49 (0)170 718-4422
johannes.specht.standards@gmail.com

Overview I

What it is

- Command Line Tool to Format YANG files
- Written in JAVA

Why?

- Tedious Tasks: Indentations, Line-wrapping (e.g. YANG description strings), etc.
- Syntax Highlighting in IEEE 802 Stds drafts for easier review
- Idea: Give IEEE 802 Stds YANG files a common design/”Look & Feel”

```
YANG file tool for IEEE 802 YANG files
  Author: Johannes Specht
Usage:
  java -jar yang802tool.jar [Options...] <YANG Input Files>
Options:
 -t N                                  : Input file tab character size.
                                         (Vorgabe: 4)
 -c WERT                               : Input file character encoding. Use
                                         encoding name 'list' to obtain a list
                                         of available character sets.
                                         (Vorgabe: UTF-8)
 -n                                    : Skip formatting the input file, just
                                         normalize. (Vorgabe: false)
 -R                                    : Enable IEEE 802 Std reference
                                         formatting (Dangerous). (Vorgabe:
                                         false)
 -D                                    : Enable IEEE 802 Std description
                                         formatting (Dangerous). (Vorgabe:
                                         false)
 -I N                                  : Output file indentation size, in
                                         numbers of characters. (Vorgabe: 2)
 -W N                                  : Output file line width limit, in
                                         number of characters. (Vorgabe: 70)
 -h                                    : Print this help and and exit.
                                         (Vorgabe: true)
 -q                                    : Omit console outputs. (Vorgabe: false)
 -o DATEI                              : Output directory. If not provided,
                                         the output YANG is written to STDOUT.
 -m                                    : If set, Maker Interchange Format
                                         (MIF) file(s) with highlightings are
                                         written into the output directory.
                                         (Vorgabe: false)
 --last-draft-yang-file DATEI          : Last draft YANG file for change
                                         indications in MIF output file(s), if
                                         such a file exists. Only used if MIF
                                         output is generated.
 --last-draft-tab-size N               : Last draft tab character size. Only
                                         used if MIF output is generated.
                                         (Vorgabe: 4)
 --last-draft-encoding WERT            : Last draft character encoding. Use
                                         encoding name 'list' to obtain a list
                                         of available character sets. Only
                                         used if MIF output is generated.
```

Overview II

Requirements

- JAVA Capable OS (e.g., Windows, Linux)
- Up-to-date Oracle JRE
- May work with other JDKs/JREs (I've never tried this)

Library Dependencies

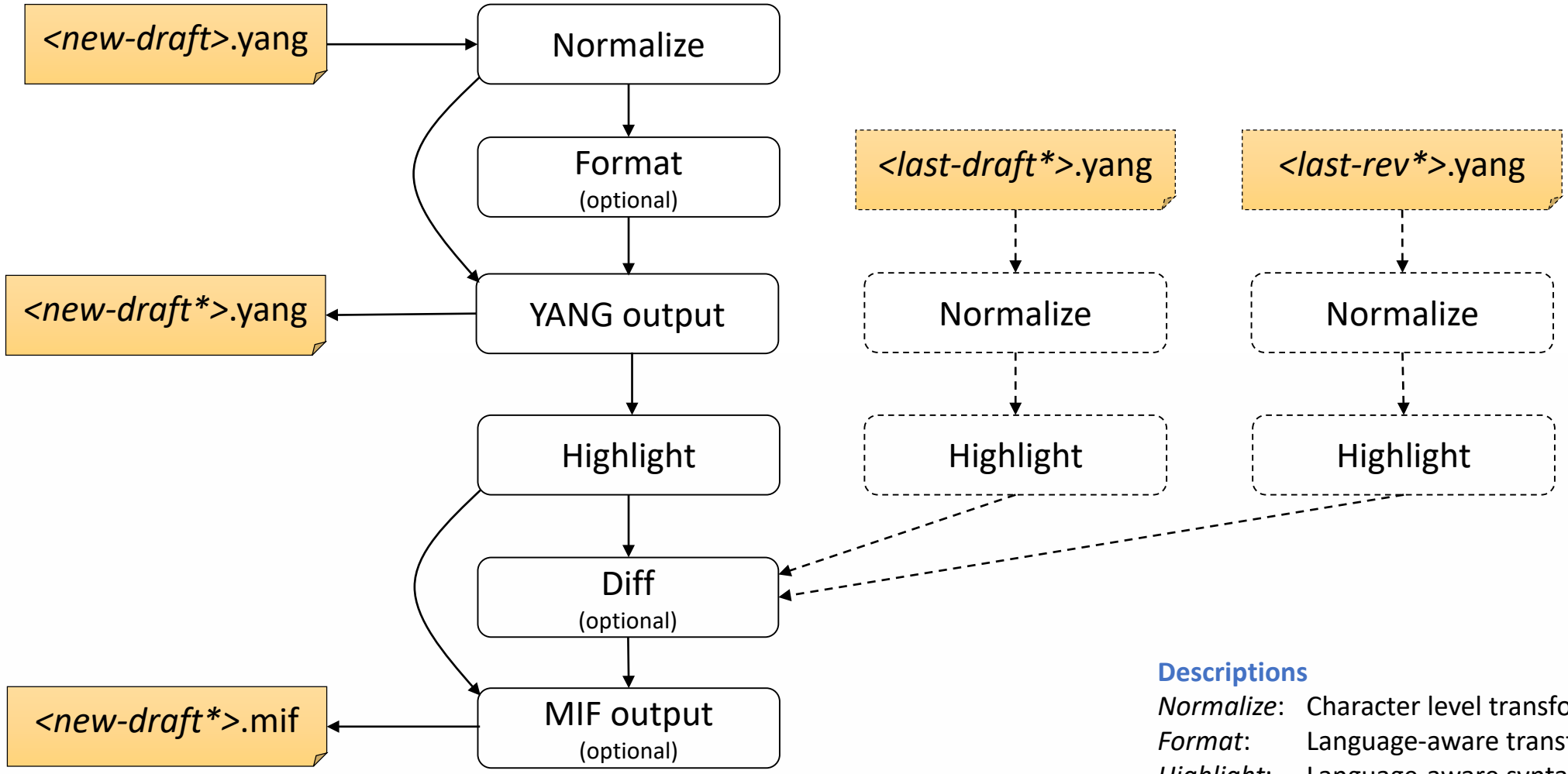
- AntLR V4, StringTemplate V4, Args4j, Apache Commons, diff-match-patch
  - Essentially Apache 2.0 and BSD licensed
- All libraries and license texts included in a single runnable JAR file

Start

```
java -jar yang802tool.jar -h
```

```
--last-draft-bars                        : If set and a last draft YANG file is
                                           provided, changes since the last
                                           draft are indicated by change bars.
                                           Only used if MIF output is generated.
                                           (Vorgabe: false)
--last-draft-edits                       : If set and a last draft YANG file is
                                           provided, changes are indicated are
                                           indicated by tracked text edits
                                           (insertions and deletions). Only used
                                           if MIF output is generated. (Vorgabe:
                                           false)
--last-draft-insert-color [Black |       : If set and a last draft YANG file is
White | Red | Green | Blue | Cyan |        provided, inserted text is set to the
Magenta | Yellow | Dark Grey | Pale      given text color. If not set, the
Green | Forest Green | Royal Blue |       default color is used. Only used if
Mauve | Light Salmon | Olive | Salmon]    MIF output is generated.
--last-draft-delete-color [Black |       : If set and a last draft YANG file is
White | Red | Green | Blue | Cyan |        provided, deleted text is set to the
Magenta | Yellow | Dark Grey | Pale      given text color. If not set, the
Green | Forest Green | Royal Blue |       default color is used. Only used if
Mauve | Light Salmon | Olive | Salmon]    MIF output is generated.
--last-rev-tab-size N                    : Last draft tab character size. Only
                                           used if MIF output is generated.
                                           (Vorgabe: 4)
--last-rev-encoding WERT                 : Last draft character encoding. Use
                                           encoding name 'list' to obtain a list
                                           of available character sets. Only
                                           used if MIF output is generated.
                                           (Vorgabe: UTF-8)
--last-rev-yang-file DATEI               : Last YANG file published by IEEE-SA
                                           for change indications (underline and
                                           strikeout), if such a file exists.
                                           Only used if MIF output is generated.
--last-rev-insert-color [Black |         : If set and a last published YANG file
White | Red | Green | Blue | Cyan |        is provided, inserted text is set to
Magenta | Yellow | Dark Grey | Pale      the given text color. Only used if
Green | Forest Green | Royal Blue |       MIF output is generated.
Mauve | Light Salmon | Olive | Salmon]
--last-rev-delete-color [Black |         : If set and a last published YANG file
White | Red | Green | Blue | Cyan |        is provided, deleted text is set to
Magenta | Yellow | Dark Grey | Pale      the given text color. Only used if
Green | Forest Green | Royal Blue |       MIF output is generated.
Mauve | Light Salmon | Olive | Salmon]
```

Program Flow



Descriptions

Normalize: Character level transformations
Format: Language-aware transformations (e.g., YANG)
Highlight: Language-aware syntax highlighting
Diff: Indication of changes between different YANG files

Normalization and Basic Formatting

```
    feature closed-gate-state {
        description
            "The bridge component supports gate state clos
        reference
            "IEEE 802.1Qcr";
    }

/* Types and groupings */
    typedef priority-spec-type {
    type enumeration {
    enum zero { value 0; description "Priority 0";}
    enum one { value 1; description "Priority 1";}
    enum two { value 2; description "Priority 2";}
      enum three {
        value 3;
        description
          "Priority 3";
      }
```

Tab replacement

Indentation fixing

"Light" Syntax Highlighting (FrameMaker)

```
feature closed-gate-state {
    description
      "The bridge component supports gate state closed.";
    reference
      "IEEE Std 802.1Qcr";
  }

  /* Types and groupings */
  typedef priority-spec-type {
    type enumeration {
      enum zero {
        value 0;
        description
          "Priority 0";
      }
      enum one {
        value 1;
        description
          "Priority 1";
      }
      enum two {
        value 2;
        description
          "Priority 2";
      }
      enum three {
        value 3;
        description
          "Priority 3";
      }
```

Not Shown

Code page fixing to UTF-8 and Line end unification ('\n')

(cmp. RFC7950, §6)

Not Shown

Re-ordering of YANG sibling nodes

NOTE: To my very best knowledge, there is no normative ordering requirement. However, it helps muting an associated PYANG "error" message.

(cmp. RFC7950, §14)

Formatting Machine Readable Strings

```
typedef stream-gate-ref {
  type leafref {
    path
      '/dot1q:bridges/dot1q:bridge/dot1q:component/sfsg:stream-gates/sfsg:stream-gate-instance-table/sfsg:stream-gate-instance-id';
  }
```

Break long paths at “reasonable” positions (readability)

```
typedef stream-gate-ref {
  type leafref {
    path
      '/dot1q:bridges'+
      '/dot1q:bridge'+
      '/dot1q:component'+
      '/sfsg:stream-gates'+
      '/sfsg:stream-gate-instance-table'+
      '/sfsg:stream-gate-instance-id';
  }
  description
    "This type is used to refer to a stream gate instance.";
}
```

Known Machine Grammars:

- YANG
  (RFC7950)
- Xpath
  (partial)
- Regular Expressions
  (“XML Schema Part 2: Datatypes Second Edition, W3C Recommendation 28 October 2004)

Description Formatting

```
description
  "An IPV can be either of the following:
     1) The null value. For a frame that passes through the gate, the priority value associated with the frame is used to determine the frame's traffic class, using the Tra
     2) An internal priority value. For a frame that passes through the gate, the IPV is used, in place of the priority value associated with the frame, to determine the fr
reference
  "IEEE 802.1Qcr, 8.6.5.2";
```

Understands list formats (numbered, lettered – cmp. “2014 IEEE-SA Standards Style Manual”, §11.3)

```
description
  "An IPV can be either of the following:
     1) The null value. For a frame that passes through the gate, the
        priority value associated with the frame is used to determine
        the frame's traffic class, using the Traffic Class Table as
        specified in 8.6.6.
     2) An internal priority value. For a frame that passes through the
        gate, the IPV is used, in place of the priority value associated
        with the frame, to determine the frame's traffic class, using
        the Traffic Class Table as specified in 8.6.6.";
reference
  "8.6.5.2 of IEEE Std 802.1Qcr";
```

Linewrapping at word boundaries ➔ Fit into Std documents

How to Use It/How I Use It

1. Execute

   java –jar *[pretty printer jar path/]***yang802tool.jar**
   "*<input file path>***"**
   -W 76 -m -D
   -o "*<output directory path>***"**

2. Copy&Paste generated Framemaker output file contents into Stds Draft

3. Generate Stds Draft PDF

4. Attach generated .YANG file to Stds Draft PDF

“diff” (new!): Overview

Character Accuracy

Googles/Neil Fraser’s Algorithm, a.k.a. “diff-match-patch”[1]

One- or Two-Level Highlighting

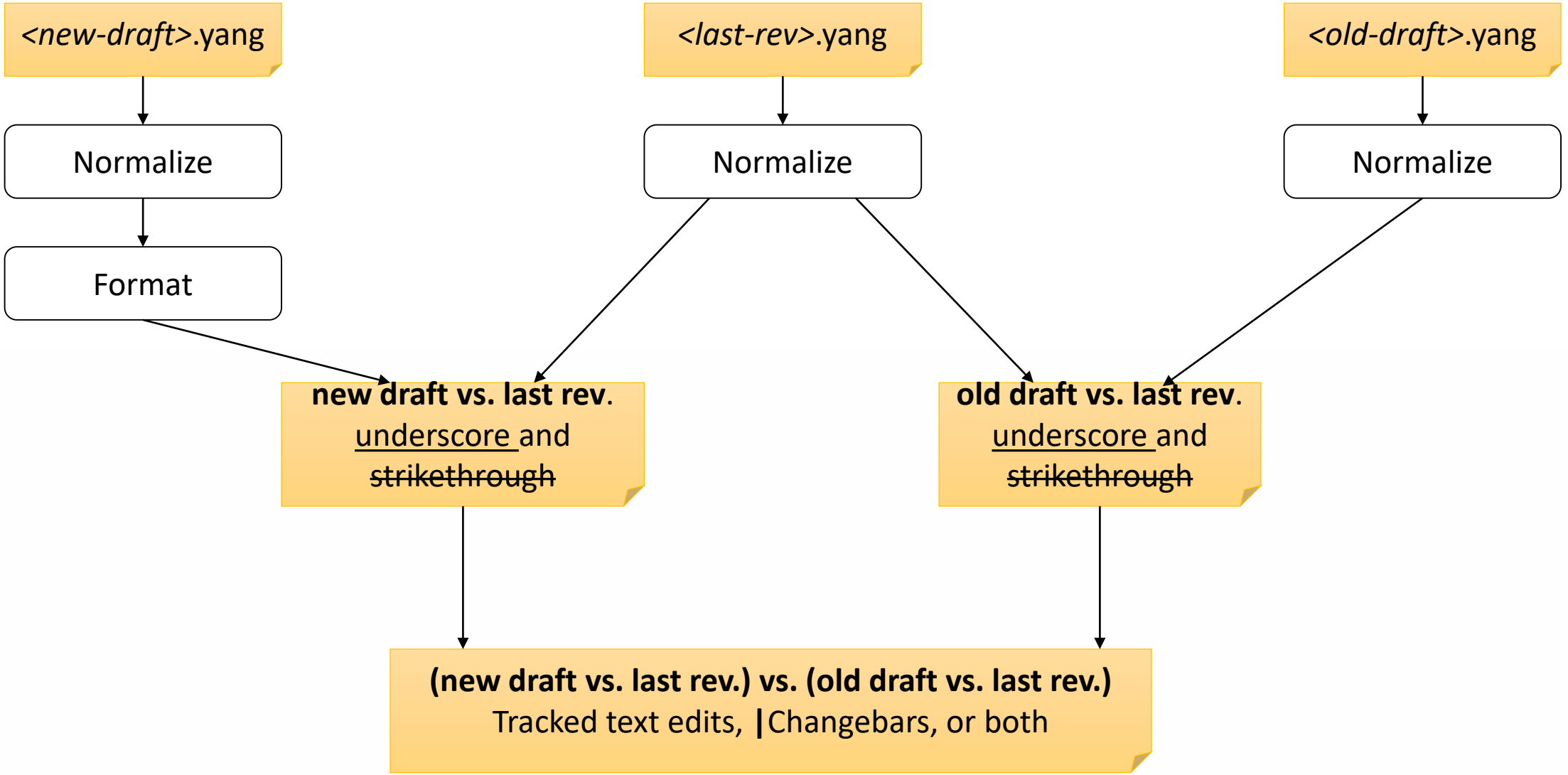
- New project’s YANG code vs. old published YANG code (revision) – Amendments and corrigenda
  (cmp. “2014 IEEE-SA Standards Style Manual”, §18.2)
- Last draft vs. new draft
- Combined (next slide)

How?

- Underscore, ~~Strikethrough~~
- ▌Changebars, conservatively placed (even font changes, e.g. regular normal to ~~Strikethrough~~)
- FrameMaker’s tracked text edits
  - Previews like “Final” and “Old” previews possible
  - System username – should I implement an override option?
- Configurable colors, style (tracked text edits, Changebars, both)

[1]: Eugene Myer’s Algorithm (1986) plus post-processing for human readability – produced the best results in my experiments

“diff” (new!): Simplified Flow



“diff” (new!): Two-Level Change Bars and Tracked Text Edits

```
  description
    "This file is a demonstration file for the YANG pretty print
    functions.";
  revision 1980-01-01 {
    description
      "Updates from an anonymous project.";
    reference
      "N/A";
  }
  revision 1970-01-01 {
    description
      "Initial revision.";
    reference
      "N/A";
  }
  typedef bridge-ref-type {
    type leafref {
      path
        '/dot1q:bridges'+
        '/dot1q:bridge'+;
    }
    '/dot1q:component';
  }
  description
    "A broken bridge refdescription
    "A working bridge reference with an exhaustive explanatior
    to the previous draft. The reason for this explanation is
```

```
  description
    "This file is a demonstration file for the YANG pretty print
    functions.";
  revision 1980-01-01 {
    description
      "Updates from an anonymous project.";
    reference
      "N/A";
  }
  revision 1970-01-01 {
    description
      "Initial revision.";
    reference
      "N/A";
  }
  typedef bridge-ref-type {
    type leafref {
      path
        '/dot1q:bridges'+
        '/dot1q:bridge'+;
    }
    '/dot1q:component';
  }
  description
    "A broken bridge referencedescription
    "A working bridge reference with an exhaustive explanatio
    to the previous draft. The reason for this explanation is
```

Caveats

Outdated IEEE Stds Reference Parser

- Understands “old” Format in YANG files, ignores “new” Format
- Worked well in earlier stages of .1Qcr, .1Qcp, etc.
- Can be bypassed (omit command line switch “-R”)

“Tolerant” Maximum Line Lengths (command line switch “-W”)

- Expect a few characters more/less for string boundary, concatenation, and termination characters (“,’,+,;)

Long Words in Descriptions

- Formatting “description” strings does not break words. Why? Think backwards – implementing breaking is easy, implementing re-merging on text modification is tough/impossible (too much semantic information needed, including [but not limited to] human languages).

“Dual-Diff-Font-Glitch”

- FrameMaker format limitation: No dual layer font overrides. In other words, switching to “Preview Old” may show text with fonts from “Final”. Consider this a temporary glitch, usually not visible in drafts (always “Final”)

Bugs

- Limited testing samples so far → there will be bugs. Please help finding them!

Proposed Next Steps

1. Provide executable JAR to IEEE 802.1 YANG editors for testing
   - Github: https://github.com/JohannesSpecht/yang802tool-pre
   - Private Repository - If you are an editor and want to test it, please contact me!
2. Fix issues
   - Based on Editors feedback
3. Collect feature and change requests for enhancements
   - Feasibility
   - Relevance
   - …
4. Publish
   1. Github?
   2. IEEE Server private/public?