Network Working Group
Internet-Draft
Intended status: Informational
Expires: June 15, 2018

R. Tse
N. Nicholas
J. Lau
P. Brasolin
Ribose

December 12, 2017

AsciiRFC: Authoring Internet-Drafts And RFCs Using AsciiDoc
draft-ribose-asciirfc-04

Abstract

This document describes the AsciiDoc syntax extension called AsciiRFC designed for authoring IETF Internet-Drafts and RFCs.

AsciiDoc is a human readable document markup language which affords more granular control over markup than comparable schemes such as Markdown.

The AsciiRFC syntax is designed to allow the author to entirely focus on text, providing the full power of the resulting RFC XML through the AsciiDoc language, while abstracting away the need to manually edit XML, including references.

This document itself was written and generated into RFC XML v2 (RFC7749) and RFC XML v3 (RFC7991) directly through asciidoctor-rfc, an AsciiRFC generator.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Internet-Drafts and RFCs intended for publication submission to the IETF can be written in a multitude of formats today, including:

- XML: RFC XML v2 [RFC7749] and v3 [RFC7991]

- nroff: through "NroffEdit" [NroffEdit]

- Microsoft Word: through usage of [RFC5385]

- Lyx: through [lyx2rfc]

- Pandoc: [RFC7328], through [pandoc2rfc] or [draftr]

- Kramdown: through [kramdown-rfc2629]

- mmark: through [mmark]

Interestingly, the last three are Markdown [RFC7763] variants.

As specified in [RFC7990], the IETF intends for the canonical format of RFCs to transition from plain-text ASCII to RFC XML v3 [RFC7991]. While plain-text will continue to be accepted from authors by the IETF, at least in the short- to medium-term, XML will be preferred for submission, and any plain-text submissions will need to be converted to RFC XML v3.

While this need is already met for RFC XML v2 [RFC7749] by the tools specified above, the transition to RFC XML v3 [RFC7991] places added onus on authors to generate compliant XML.

[AsciiDoc] is an alternative markup language to Markdown, with features that make it attractive as a markup language for RFC with XML output. This document describes the use of [Asciidoctor], a Ruby-based enhancement of the original AsciiDoc markup language, for RFC XML markup, with a Ruby gem written by the authors used to render Asciidoctor documents as RFC XML. The markup language used specifically for the purpose of generating RFC XML document is called "AsciiRFC".

Section 1.2 of [RFC7764] famously states that "there is no such thing as "invalid" Markdown, there is no standard demanding adherence to the Markdown syntax, and there is no governing body that guides or impedes its development." While there are contexts where that lack of rigour is helpful, the authoring of RFCs does have a standard and a governing body, and there is such a thing as invalid RFC XML. A more rigorous counterpart to Markdown, which still preserves its basic approach to formatting, is useful in generating RFC XML that encompasses a fuller subset of the specification, and preempting malformed RFC XML output.

Compared to Markdown [Asciidoctor-Manual],

- AsciiDoc was designed from the beginning as a publishing language: it was initially intended as a plain-text alternative to the DocBook XML schema. For that reason, Asciidoctor natively supports the full range of formatting required by RFC XML (including notes, tables, bibliographies, source-code blocks, and definition lists), without resorting to embedded HTML or Markdown "flavours".

- AsciiDoc in its Ruby-based Asciidoctor implementation is extensible, with a well-defined API. (Extensions have been harnessed to deal with bibliographic preprocessing for AsciiRFC.)

- AsciiRFC allows granular control of rendering, including user-specified attributes of text blocks.

- The Asciidoctor implementation allows document inclusion, for managing large-scale documentation projects.

- AsciiRFC allows granular control of permutations of block nesting, such as source code within lists or definition lists within unordered lists.

- As a more formal counterpart to Markdown, AsciiDoc is well-suited to generating XML that needs to conform to a specified schema.

As with Markdown, there is a wide range of tools that can render AsciiDoc; so AsciiRFC drafts of RFC documents can be previewed and accessed without depending on the RFC tools ecosystem. Our realisation of RFC XML in AsciiRFC has aimed to ensure that, as much as possible, the markup language can be can be processed by generic Asciidoctor tools. (The only exception to this as an add-on is the optional bibliography module, which allows bibliographies to be assembled on the fly based on citations in a document: see Section 17.2.)

2. Conventions Used in This Document

The key words "*MUST*", "*MUST NOT*", "*REQUIRED*", "*SHALL*", "*SHALL NOT*", "*SHOULD*", "*SHOULD NOT*", "*RECOMMENDED*", "*MAY*", and "*OPTIONAL*" in this document are to be interpreted as described in [RFC2119].

2.1. Definitions

In this document, _AsciiDoc_ refers to the markup language generically. _Asciidoctor_ refers specifically to the Ruby-based implementation of the markup language, which has enhanced the original markup language. The RFC XML document converter contributed by the authors uses a subset of _Asciidoctor_, with some minor additions (a few document attributes specific to RFC XML, some macros specific to citation processing, and some templated use of _Asciidoctor_ crossreferences). This variant of _Asciidoctor_ markup is referred to as _AsciiRFC_.

3. Document Structure And AsciiDoctor Syntax

The syntax of Asciidoctor is presented in the Asciidoctor user manual [Asciidoctor-Manual]. AsciiRFC is a subset of Asciidoctor syntax, with the addition of bibliographic macros (Section 17.2).

Asciidoctor consists of:

- A document header, containing a title, a list of authors, and document attributes in lines prefixed with ":".

- An optional document preamble, separated from document header by a blank line.

- A number of sections, set off by a section title (a line prefixed with two or more "=".

A section may contain:

- Other sections, whose level of nesting is indicated by the number of "=" in their header.

- Blocks of text. Blocks can have metadata (including a title, an anchor for cross-references, and attributes.)

Blocks can be:

- Paragraphs, which are terminated by blank lines.

- Lists. List items are by default paragraphs, but can span over multiple paragraphs.

- Delimited blocks (with a line delimiter on either side of them); these include tables, notes, sidebars, source code, block quotes, examples, and unprocessed content (e.g. raw XML). Delimited blocks contain by default one or more paragraphs.

- List items can contain other blocks, including both nested lists and delimited blocks.

- Some delimited blocks can contain other delimited blocks; for example, examples can contain source code as well as discussion in paragraphs.

- Blocks of text consist of inline text, which themselves can contain markup.

Inline markup includes:

- Text formatting: bold, italic, superscript, subscript, monospace.

- Custom markup macros. (AsciiRFC uses one: "bcp14".)

- URLs, including display text.

- Inline anchors.

- Cross-references to anchors (IDs of blocks or spans of text), including display text.

- Images, audio, and visual files. (AsciiRFC only supports images.)

- Index terms.

- Equations (native support for [AsciiMathML] and [TeX-LaTeX], via the [MathJax] tool). (Not supported in AsciiRFC, since there is no RFC XML equivalent.)

- Footnotes. (Not supported in AsciiRFC.)

3.1. AsciiRFC Mapping To Asciidoctor

The Asciidoctor document structure aligns with the RFC XML v2 and v3 structure. In the following, v3 equivalences are given.

Header
"<rfc>" attributes, most "front" elements.

Preamble
"front/abstract" and "front/note".

Sections
"middle/section" elements.

Sections with bibliography style attributes
"back/references" elements.

Sections with appendix style attributes
"back/section" elements.

Paragraphs
"t" elements.

Lists
"ul", "ol", "dl" elements.

Delimited blocks
"artwork", "aside", "blockquote", "figure", "note", "sourcecode", "table".

Inline markup
"bcp14", "br", "cref", "em", "eref", "iref", "relref", "strong", "sub", "sup", "tt", "xref".

Full details of the mapping of AsciiRFC elements to RFC XML v2 and v3 elements, and of how to convert AsciiRFC documents to RFC XML, are given in the documentation of [asciidoctor-rfc].

3.2. Simple Illustration

This section gives an overview of how to create an RFC XML document in AsciiRFC, with some pitfalls to be aware of.

Illustrations are in RFC XML v3, although the converter deals with both versions of RFC XML.

A sample AsciiRFC document is provided in Figure 1, and its corresponding rendering in:

- RFC XML v3 (Figure 2)

- RFC XML v2 (Figure 3)

```
= Four Yorkshiremen Sketch
Tim Brooke-Taylor; John Cleese; Graham Chapman; Marty Feldman
:doctype: internet-draft
:abbrev: 4 Yorkshiremen
:obsoletes: 10, 120
:updates: 2010, 2120
:status: informational
:name: draft-four-yorkshiremen-00
:ipr: trust200902
:area: Internet
:workgroup: Network Working Group
:keyword: yorkshire, memory
:revdate: 1990-04-01T00:00:00Z
:organization: BBC
:phone: (555) 555-5555
:uri: http://example.com
:street: 10 Moulton Street
:city: Cambridge
:code: MA 02238
:email: tbt@example.com
:email_2: jc@example.com
:email_3: gc@example.com
:email_4: mf@bcc.co.uk
:smart-quotes: false
:link: https://en.wikipedia.org/wiki/Four_Yorkshiremen_sketch

[abstract]
The sketch is a parody of nostalgic conversations about humble
beginnings or difficult childhoods, featuring four men from Yorkshire
who reminisce about their upbringing. As the conversation progresses
they try to outdo one another, and their accounts of deprived
childhoods become increasingly absurd. <<michaelpalin>> <<ericidle>>
```

```
NOTE: See also Wikipedia summary

[#michaelpalin]
== Claim: Michael Palin
You were lucky. We lived for three months in a brown paper bag in a
septic tank. We used to have to get up at six o'clock in the morning,
clean the bag, eat a crust of stale bread, go to work down mill for
fourteen hours a day week in-week out. When we got home, our Dad would
thrash us to sleep with his belt! <<RFC7253>>

=== Response: Graham Chapman
Luxury. We used to have to get out of the lake at three o'clock in
the morning, clean the lake, eat a handful of hot gravel, go to work
at the mill every day for tuppence a month, come home, and Dad would
beat us around the head and neck with a broken bottle, if we were
*lucky*!

=== Response: Terry Gilliam
Well we had it tough. We used to have to get up out of the shoebox at
twelve o'clock at night, and *lick* the road clean with our tongues. We
had half a handful of freezing cold gravel, worked twenty-four hours
a day at the mill for fourpence every six years, and when we got home,
our Dad would slice us in two with a bread knife.

[#ericidle]
=== Response: Eric Idle
Right.

I had to get up in the morning at ten o'clock at night, half
an hour before I went to bed, (_pause for laughter_), eat a lump
of cold poison, work twenty-nine hours a day down mill, and pay mill
owner for permission to come to work, and when we got home,
our Dad would kill us, and dance about on our graves
singing "Hallelujah."

[bibliography]
== Normative References
++++
<reference anchor='RFC7253'
  target='https://tools.ietf.org/html/rfc7253'>
  <front>
    <title>Guidelines for Writing an IANA Considerations
      Section in RFCs</title>
    <author initials="T." surname="Krovetz">
      <organization>Sacramento State</organization>
    </author>
    <author initials="P." surname="Rogaway">
      <organization>UC Davis</organization>
```

```
    </author>
    <date month='May' year='2014'/>
  </front>
  <seriesInfo name="RFC" value="7253"/>
</reference>
++++

[appendix]
== Addendum
But you try and tell the young people today that...
and they won't believe ya.
```

Figure 1: Sample Internet-Draft in AsciiRFC

The first block of text, from "= Four Yorkshiremen Sketch" through to ":link: https://en.wikipedia.org/wiki/Four_Yorkshiremen_sketch", is the document header. It contains a title in the first line, an author attribution, and then a set of document attributes, conveying information about the document as well as information about its authors. This information ends up either as attributes of the root "rfc" tag, elements of the "front" tag, or processing instructions.

The following blocks of text, up until the first section header ("== Claim: Michael Palin"), are the document preamble. They are treated by the document converter as containing the document abstract ("abstract"), followed by any notes ("note", identified above by the "NOTE:" heading).

The first section header ("== Claim: Michael Palin") is preceded by an anchor for that section ("[#michaelpalin]"). There is a cross-reference to that anchor already in place in the abstract ("<<michaelpalin>>"). The document converter treats the first section of the document as the start of the "middle" section of the document.

The first section header is followed by a paragraph, and other sections and paragraphs. The number of "=" signs are one higher than the initial section header, which indicates that they are subsections of that section. The paragraphs contains some inline formatting (italics: "_pause for laughter_"; boldface: "*lick*"). The first paragraph also contains a citation of a reference, which in this version of AsciiRFC is treated identically to a cross-reference ("<<RFC7253>>"). (If the bibliography preprocessor were used, it would be encoded differently.)

The second last section is tagged with the style attribute "[bibliography]", which identifies it as a references container; the document converter accordingly inserts this into the "back" element

of the document. The contents of the references section are in this instance raw XML, delimited as a passthrough block (with "++++"), which the converter does not alter. The final section is tagged with the style attribute "[appendix]", and is treated as such.

The RFC XML v3 document generated from this AsciiRFC document is:

```
<?xml version="1.0" encoding="US-ASCII"?>
<!DOCTYPE rfc SYSTEM "rfc2629.dtd">
<rfc ipr="trust200902" obsoletes="10, 120" updates="2010, 2120"
    submissionType="IETF" prepTime="2017-11-25T09:54:54Z" version="3">
  <link href="https://en.wikipedia.org/wiki/Four_Yorkshiremen_sketch"/>
  <front>
    <title abbrev="4 Yorkshiremen">Four Yorkshiremen Sketch</title>
    <seriesInfo name="Internet-Draft" status="informational"
      stream="IETF" value="draft-four-yorkshiremen-00" />
    <author fullname="Tim Brooke-Taylor" surname="Brooke-Taylor">
      <organization>BBC</organization>
      <address>
        <postal>
          <street>10 Moulton Street</street>
          <city>Cambridge</city>
          <code>MA 02238</code>
        </postal>
        <phone>(555) 555-5555</phone>
        <email>tbt@example.com</email>
        <uri>http://example.com</uri>
      </address>
    </author>
    <author fullname="John Cleese" surname="Cleese">
      <address>
        <email>jc@example.com</email>
      </address>
    </author>
    <author fullname="Graham Chapman" surname="Chapman">
      <address>
        <email>gc@example.com</email>
      </address>
    </author>
    <author fullname="Marty Feldman" surname="Feldman">
      <address>
        <email>mf@bcc.co.uk<email>
      </address>
    </author>
    <date day="1" month="April" year="1990" />
    <area>Internet<area>
    <workgroup>Network Working Group</workgroup>
    <keyword>yorkshire<keyword>
```

```
    <keyword>memory<keyword>
    <abstract>
      <t>The sketch is a parody of nostalgic conversations about humble
      beginnings or difficult childhoods, featuring four men from
      Yorkshire who reminisce about their upbringing. As the
      conversation progresses they try to outdo one another, and their
      accounts of deprived childhoods become increasingly absurd.
      <xref target="michaelpalin" />
      <xref target="ericidle" /></t>
    </abstract>
    <note>
      <t>See also Wikipedia summary<t>
    </note>
  </front>
  <middle>
    <section anchor="michaelpalin" numbered="false">
      <name>Claim: Michael Palin<name>
      <t>You were lucky. We lived for three months in a brown paper bag
        in a septic tank. We used to have to get up at six o'clock in
        the morning, clean the bag, eat a crust of stale bread, go to
        work down mill for fourteen hours a day week in-week out. When
        we got home, our Dad would thrash us to sleep with his belt!
        <xref target="RFC7253" /></t>
      <section anchor="_response_graham_chapman" numbered="false">
        <name>Response: Graham Chapman<name>
        <t>Luxury. We used to have to get out of the lake at three
          o'clock in the morning, clean the lake, eat a handful of hot
          gravel, go to work at the mill every day for tuppence a month,
          come home, and Dad would beat us around the head and neck with
          a broken bottle, if we were <strong>lucky</strong>!</t>
      </section>
      <section anchor="_response_terry_gilliam" numbered="false">
        <name>Response: Terry Gilliam<name>
        <t>Well we had it tough. We used to have to get up out of the
          shoebox at twelve o'clock at night, and <strong>lick<strong>
          the road clean with our tongues. We had half a handful of
          freezing cold gravel, worked twenty-four hours a day at the
          mill for fourpence every six years, and when we got home,
          our Dad would slice us in two with a bread knife.</t>
      </section>
      <section anchor="ericidle" numbered="false">
        <name>Response: Eric Idle<name>
        <t>Right.<t>
        <t>I had to get up in the morning at ten o'clock at night, half
          an hour before I went to bed, (<em>pause for laughter</em>),
          eat a lump of cold poison, work twenty-nine hours a day down
          mill, and pay mill owner for permission to come to work, and
          when we got home, our Dad would kill us, and dance about on
```

```
          our graves singing "Hallelujah."</t>
      </section>
    </section>
  </middle>
  <back>
    <references anchor="_normative_references">
      <name>Normative References<name>
      <reference anchor="RFC7253"
          target="https://tools.ietf.org/html/rfc7253">
        <front>
          <title>Guidelines for Writing an IANA Considerations
            Section in RFCs<title>
          <author initials="T." surname="Krovetz">
            <organization>Sacramento State<organization>
          </author>
          <author initials="P." surname="Rogaway">
            <organization>UC Davis<organization>
          </author>
          <date month="May" year="2014" />
        </front>
        <seriesInfo name="RFC" value="7253" />
      </reference>
    </references>
    <section anchor="_addendum" numbered="false">
      <name>Addendum<name>
      <t>But you try and tell the young people today that&#8230;&#8203;
        and they won't believe ya'.<t>
    </section>
  </back>
</rfc>
```

Figure 2: Sample Internet-Draft In AsciiRFC, Output In RFC XML v3 Format

Some default processing instructions have already been prefixed to the XML.

Our AsciiRFC converter can also generate RFC XML v2 from the same source AsciiRFC, as shown in Figure 3. Output in RFC XML v2 is not extensively described in this document.

```
<rfc ipr="trust200902" obsoletes="10, 120" updates="2010, 2120"
  category="info" submissionType="IETF"
  docName="draft-four-yorkshiremen-00">
  <front>
    <title abbrev="4 Yorkshiremen">Four Yorkshiremen Sketch<title>
    <author fullname="Tim Brooke-Taylor" surname="Brooke-Taylor">
      <organization>BBC</organization>
```

```
      <address>
        <postal>
          <street>10 Moulton Street</street>
          <city>Cambridge</city>
          <code>MA 02238</code>
        </postal>
        <phone>(555) 555-5555</phone>
        <email>tbt@example.com</email>
        <uri>http://example.com</uri>
      </address>
    </author>
    <author fullname="John Cleese" surname="Cleese">
      <address>
        <email>jc@example.com</email>
      </address>
    </author>
    <author fullname="Graham Chapman" surname="Chapman">
      <address>
        <email>gc@example.com</email>
      </address>
    </author>
    <author fullname="Marty Feldman" surname="Feldman">
      <address>
        <email>mf@bcc.co.uk</email>
      </address>
    </author>
    <date day="1" month="April" year="1990" />
    <area>Internet</area>
    <workgroup>Network Working Group</workgroup>
    <keyword>yorkshire</keyword>
    <keyword>memory</keyword>
    <abstract>
      <t>The sketch is a parody of nostalgic conversations about humble
      beginnings or difficult childhoods, featuring four men from
      Yorkshire who reminisce about their upbringing. As the
      conversation progresses they try to outdo one another, and their
      accounts of deprived childhoods become increasingly absurd.
      <xref target="michaelpalin" />
      <xref target="ericidle" /></t>
    </abstract>
    <note title="NOTE">
      <t>See also Wikipedia summary</t>
    </note>
  </front>
  <middle>
    <section anchor="michaelpalin" title="Claim: Michael Palin">
      <t>You were lucky. We lived for three months in a brown paper bag
        in a septic tank. We used to have to get up at six o'clock in
```

```
        the morning, clean the bag, eat a crust of stale bread, go to
        work down mill for fourteen hours a day week in-week out. When
        we got home, our Dad would thrash us to sleep with his belt!
        <xref target="RFC7253" /></t>
      <section anchor="_response_graham_chapman"
          title="Response: Graham Chapman">
        <t>Luxury. We used to have to get out of the lake at three
          o'clock in the morning, clean the lake, eat a handful of hot
          gravel, go to work at the mill every day for tuppence a month,
          come home, and Dad would beat us around the head and neck with
          a broken bottle, if we were
          <spanx style="strong">lucky</spanx>!</t>
      </section>
      <section anchor="_response_terry_gilliam"
          title="Response: Terry Gilliam">
        <t>Well we had it tough. We used to have to get up out of the
          shoebox at twelve o'clock at night, and
          <spanx style="strong">lick<spanx>
          the road clean with our tongues. We had half a handful of
          freezing cold gravel, worked twenty-four hours a day at the
          mill for fourpence every six years, and when we got home,
          our Dad would slice us in two with a bread knife.</t>
      </section>
      <section anchor="ericidle" title="Response: Eric Idle">
        <t>Right.</t>
        <t>I had to get up in the morning at ten o'clock at night, half
          an hour before I went to bed, (<spanx style="emph">pause
          for laughter</spanx>),
          eat a lump of cold poison, work twenty-nine hours a day down
          mill, and pay mill owner for permission to come to work, and
          when we got home, our Dad would kill us, and dance about on
          our graves singing "Hallelujah."</t>
      </section>
    </section>
  </middle>
  <back>
    <references title="Normative References">
      <reference anchor="RFC7253"
          target="https://tools.ietf.org/html/rfc7253">
        <front>
          <title>Guidelines for Writing an IANA Considerations
            Section in RFCs</title>
          <author initials="T." surname="Krovetz">
            <organization>Sacramento State</organization>
          </author>
          <author initials="P." surname="Rogaway">
            <organization>UC Davis</organization>
          </author>
```

```
          <date month="May" year="2014" />
        </front>
        <seriesInfo name="RFC" value="7253" />
      </reference>
    </references>
    <section anchor="_addendum" title="Addendum">
      <t>But you try and tell the young people today that&#8230;&#8203;
        and they won't believe ya'.</t>
    </section>
  </back>
</rfc>
```

Figure 3: Sample Internet-Draft In AsciiRFC, Output In RFC XML v2 Format

4. Header And Document Attributes

The header gives the document title, followed by an optional author attribution, and a series of document attributes, with no carriage return breaks.

4.1. Title And Basic Attributes

Document attributes are used to populate attributes of the root "rfc" element, "front" elements, and document-level processing instructions.

- ":doctype:" determines whether the document will be considered "rfc" or "internet-draft", and interprets other attributes accordingly.

- Certain attributes ("workgroup", "area", "keyword") are comma delimited, and result in repeated RFC XML elements.

Figure 4 demonstrates how to set the document header in AsciiRFC, with its rendering in v3 shown in Figure 5.

```
   = Four Yorkshiremen Sketch
   Tim Brooke-Taylor <tbt@example.com>
   :doctype: internet-draft
   :abbrev: 4 Yorkshiremen
   :obsoletes: 10, 120
   :updates: 2010, 2120
   :status: informational
   :name: draft-four-yorkshiremen-00
   :ipr: trust200902
   :area: Internet
   :workgroup: Network Working Group
   :keyword: yorkshire, memory
   :revdate: 1990-04-01T00:00:00Z
```

Figure 4: AsciiRFC Document Header

```
  <rfc ipr="trust200902" obsoletes="10, 120" updates="2010, 2120"
      submissionType="IETF" prepTime="2017-11-25T10:13:46Z" version="3">
    <front>
      <title abbrev="4 Yorkshiremen">Four Yorkshiremen Sketch</title>
      <seriesInfo name="Internet-Draft" status="informational"
          stream="IETF" value="draft-four-yorkshiremen-00" />
      <author fullname="Tim Brooke-Taylor" surname="Brooke-Taylor">
        <address>
          <email>tbt@example.com</email>
        </address>
      </author>
      <date day="1" month="April" year="1990" />
      <area>Internet</area>
      <workgroup>Network Working Group</workgroup>
      <keyword>yorkshire</keyword>
      <keyword>memory</keyword>
```

Figure 5: AsciiRFC Document Header Rendered As RFC XML v3

4.2. Detailed Author Information

The document header can spell out further information about authors, including contact details. The AsciiRFC header is shown in Figure 6 with its rendering in RFC XML v3 shown in Figure 7.

```
= Four Yorkshiremen Sketch
Tim Brooke-Taylor <tbt@example.com>
:doctype: internet-draft
:abbrev: 4 Yorkshiremen
:obsoletes: 10, 120
:updates: 2010, 2120
:status: informational
:name: draft-four-yorkshiremen-00
:ipr: trust200902
:area: Internet
:workgroup: Network Working Group
:keyword: yorkshire, memory
:revdate: 1990-04-01T00:00:00Z
:organization: BBC
:phone: (555) 555-5555
:uri: http://bbn.com
:street: 10 Moulton Street
:city: Cambridge
:code: MA 02238
```

Figure 6: AsciiRFC Document Header With One Author

```
<rfc ipr="trust200902" obsoletes="10, 120" updates="2010, 2120"
    submissionType="IETF" prepTime="2017-11-25T10:15:02Z" version="3">
  <front>
    <title abbrev="4 Yorkshiremen">Four Yorkshiremen Sketch</title>
    <seriesInfo name="Internet-Draft" status="informational"
        stream="IETF" value="draft-four-yorkshiremen-00" />
    <author fullname="Tim Brooke-Taylor" surname="Brooke-Taylor">
      <organization>BBC</organization>
      <address>
        <postal>
          <street>10 Moulton Street</street>
          <city>Cambridge</city>
          <code>MA 02238</code>
        </postal>
        <phone>(555) 555-5555</phone>
        <email>tbt@example.com</email>
        <uri>http://bbn.com</uri>
      </address>
    </author>
    <date day="1" month="April" year="1990" />
    <area>Internet</area>
    <workgroup>Network Working Group</workgroup>
    <keyword>yorkshire</keyword>
    <keyword>memory</keyword>
```

Figure 7: AsciiRFC Document Header With One Author (RFC XML v3)

Details of a second, third etc. author, including their organization and contact details, are provided by suffixing the relevant author attributes with "_2", "_3" etc., as shown in Figure 8 and its v3 rendering Figure 9.

```
= Four Yorkshiremen Sketch
Tim Brooke-Taylor <tbt@example.com>; John Cleese <jc@example.com>
:doctype: internet-draft
:status: informational
:name: draft-four-yorkshiremen-00
:ipr: trust200902
:organization: BBC
:phone: (555) 555-5555
:uri: http://example.com
:street: 10 Moulton Street
:city: Cambridge
:code: MA 02238
:forename_initials: T.
:lastname: Brooke-Taylor
:street: 12 Moulton Street
:city: London
:country: United Kingdom
:forename_initials_2: J.
:lastname_2: Cleese
:uri_2: https://twitter.com/johncleese
```

Figure 8: AsciiRFC Document Header With Multiple Authors

```
<rfc ipr="trust200902" submissionType="IETF"
    prepTime="2017-11-25T10:19:32Z" version="3">
  <front>
    <title>Four Yorkshiremen Sketch</title>
    <seriesInfo name="Internet-Draft" status="informational"
        stream="IETF" value="draft-four-yorkshiremen-00" />
    <author fullname="Tim Brooke-Taylor"
        surname="Brooke-Taylor" initials="T.">
      <organization>BBC</organization>
      <address>
        <postal>
          <street>12 Moulton Street</street>
          <city>London</city>
          <code>MA 02238</code>
          <country>United Kingdom</country>
        </postal>
        <phone>(555) 555-5555</phone>
        <email>tbt@example.com</email>
        <uri>http://example.com</uri>
      </address>
    </author>
    <author fullname="John Cleese" surname="Cleese" initials="J.">
      <address>
        <email>jc@example.com</email>
        <uri>https://twitter.com/johncleese</uri>
      </address>
    </author>
    <date day="25" month="November" year="2017" />
```

Figure 9: AsciiRFC Document Header With Multiple Authors (RFC XML v3)

The initial author attribution in AsciiRFC, e.g. "Tim Brooke-Taylor <tbt@bbc.co.uk>; John Cleese <jc@bbc.co.uk>" in the example above, expects a strict format of First Name, zero or more Middle Names, Last name, and cannot process honorifics like "Dr." or suffixes like "Jr.".

Name attributes with any degree of complexity should be overriden by using the ":fullname:" and ":lastname:" attributes. The AsciiRFC ":forename_initials:" attribute replaces the built-in Asciidoctor ":initials:" attribute (which includes the surname initial), and is not automatically populated from the name attribution.

4.3. XML Processing Information

A document header may also contain attribute headers which are treated as XML processing instructions. An AsciiRFC example is shown in Figure 10 with its rendering in Figure 11.

```
= Four Yorkshiremen Sketch
Tim Brooke-Taylor <tbt@example.com>
:doctype: internet-draft
:status: informational
:name: draft-four-yorkshiremen-00
:ipr: trust200902
:revdate: 1990-04-01T00:00:00Z
:rfcedstyle: yes
:text-list-symbols: yes
:rfc2629xslt: true
```

Figure 10: AsciiRFC Document Header With XML Processing Information

```
<rfc ipr="trust200902" submissionType="IETF"
    prepTime="2017-11-25T10:21:56Z" version="3">
  <front>
    <title>Four Yorkshiremen Sketch</title>
    <seriesInfo name="Internet-Draft" status="informational"
        stream="IETF" value="draft-four-yorkshiremen-00" />
    <author fullname="Tim Brooke-Taylor" surname="Brooke-Taylor">
      <address>
        <email>tbt@example.com</email>
      </address>
    </author>
    <date day="1" month="April" year="1990" />
```

Figure 11: AsciiRFC Document Header With XML Processing Information (RFC XML v3)

4.4. AsciiRFC-specific Document Attributes

A few document attributes are specific to the operation of the RFC XML document converter:

:no-rfc-bold-bcp14: false
: overrides the wrapping by default of boldface uppercase BCP14 [RFC2119] words (e.g. "*MUST NOT*") with the "bcp14" element.

:smart-quotes: false
: overrides Asciidoctor's conversion of straight quotes and apostrophes to smart quotes and apostrophes.

:inline-definition-lists: true
: overrides the RFC XML v2 "idnits" requirement that a blank line be inserted between a definition list term and its definition.

```
= Four Yorkshiremen Sketch
Tim Brooke-Taylor <tbt@example.com>
:doctype: internet-draft
:status: informational
:name: draft-four-yorkshiremen-00

== Section 1
The specification *MUST NOT* use the word _doesn't_.
```

Figure 12: AsciiRFC Document Header Without RFC-specific Attributes

```
<rfc submissionType="IETF" prepTime="2017-11-25T10:23:39Z" version="3">
  <front>
    <title>Four Yorkshiremen Sketch</title>
    <seriesInfo name="Internet-Draft" status="informational"
        stream="IETF" value="draft-four-yorkshiremen-00" />
    <author fullname="Tim Brooke-Taylor" surname="Brooke-Taylor">
      <address>
        <email>tbt@example.com</email>
      </address>
    </author>
    <date day="25" month="November" year="2017" />
  </front>
  <middle>
    <section anchor="_section_1" numbered="false">
      <name>Section 1</name>
      <t>The specification  <bcp14>MUST NOT</bcp14>
        use the word <em> doesn&#8217;t</em>.</t>
    </section>
  </middle>
</rfc>
```

Figure 13: AsciiRFC Document Header Without RFC-specific Attributes (RFC XML v3)

```
= Four Yorkshiremen Sketch
Tim Brooke-Taylor <tbt@example.com>
:doctype: internet-draft
:status: informational
:name: draft-four-yorkshiremen-00
:no-rfc-bold-bcp14: false
:smart-quotes: false

== Section 1
The specification *MUST NOT* use the word _doesn't_.
```

Figure 14: AsciiRFC Document Header With Overridden RFC-specific Attributes

```
<rfc submissionType="IETF" prepTime="2017-11-25T10:23:39Z" version="3">
  <front>
    <title>Four Yorkshiremen Sketch</title>
    <seriesInfo name="Internet-Draft" status="informational"
        stream="IETF" value="draft-four-yorkshiremen-00" />
    <author fullname="Tim Brooke-Taylor" surname="Brooke-Taylor">
      <address>
        <email>tbt@example.com</email>
      </address>
    </author>
    <date day="25" month="November" year="2017" />
  </front>
  <middle>
    <section anchor="_section_1" numbered="false">
      <name>Section 1</name>
      <t>The specification <strong>MUST NOT</strong>
        use the word <em>doesn't</em>.</t>
    </section>
  </middle>
</rfc>
```

Figure 15: AsciiRFC Document Header With Overridden RFC-specific Attributes (RFC XML v3)

5. Preamble

The preamble in AsciiRFC is the text between the end of the document header (which terminates with a blank line) and the first section of text.

Any paragraphs of text in the preamble are treated as an abstract, and may optionally be tagged with the "abstract" style attribute.

Any notes in the preamble are treated as a "note" element.

An example of setting the preamble is given in Figure 16 with its rendering in Figure 17.

```
= Four Yorkshiremen Sketch
Tim Brooke-Taylor <tbt@example.com>
:doctype: internet-draft
:status: informational
:name: draft-four-yorkshiremen-00

The "Four Yorkshiremen" sketch is a comedy sketch written by
Tim Brooke-Taylor, John Cleese, Graham Chapman and Marty Feldman and
originally performed on their TV series _At Last the 1948 Show_ in 1967.
It later became associated with the comedy group Monty Python
(which included Cleese and Chapman), who performed it in their live
shows, including _Monty Python Live at the Hollywood Bowl_.

The sketch is a parody of nostalgic conversations about humble
beginnings or difficult childhoods, featuring four men from Yorkshire
who reminisce about their upbringing. As the conversation progresses
they try to outdo one another, and their accounts of deprived
childhoods become increasingly absurd.

NOTE:  Barry Cryer is the wine waiter in the original performance
and may have contributed to the writing.

[NOTE]
.Original Recording
==
The original performance of the sketch by the four creators is one of
the surviving sketches from the programme and can be seen on the
_At Last the 1948 Show_ DVD.
==
```

Figure 16: AsciiRFC With Preamble

```
<rfc submissionType="IETF" prepTime="2017-11-25T10:32:27Z" version="3">
  <front>
    <title>Four Yorkshiremen Sketch</title>
    <seriesInfo name="Internet-Draft" status="informational"
        stream="IETF" value="draft-four-yorkshiremen-00" />
    <author fullname="Tim Brooke-Taylor" surname="Brooke-Taylor">
      <address>
        <email>tbt@example.com</email>
      </address>
    </author>
    <date day="25" month="November" year="2017" />
    <abstract>
      <t>The "Four Yorkshiremen" sketch is a comedy sketch written by
        Tim Brooke-Taylor, John Cleese, Graham Chapman and Marty Feldman
        and originally performed on their TV series <em>At Last the 1948
        Show</em> in 1967. It later became associated with the comedy
        group Monty Python (which included Cleese and Chapman), who
        performed it in their live shows, including <em>Monty Python
        Live at the Hollywood Bowl</em>.</t>
      <t>The sketch is a parody of nostalgic conversations about humble
        beginnings or difficult childhoods, featuring four men from
        Yorkshire who reminisce about their upbringing. As the
        conversation progresses they try to outdo one another, and their
        accounts of deprived childhoods become increasingly absurd.</t>
    </abstract>
    <note>
      <t>Barry Cryer is the wine waiter in the original performance and
        may have contributed to the writing.</t>
    </note>
    <note>
      <name>Original Recording</name>
      <t>The original performance of the sketch by the four
        creators is one of the surviving sketches from the programme
        and can be seen on the <em>At Last the 1948 Show</em> DVD.</t>
    </note>
  </front>
```

Figure 17: AsciiRFC With Preamble (RFC XML v3)

6. Sections and Paragraphs

Section headers are given with a sequence of "=", the number of "=" giving the header level. Section numbering is toggled with the in-document attribute ":sectnums:" (on), ":sectnums!:" (off). The "toc" attribute can also be set on sections, indicating whether the section can be included in the document's table of contents.

Figure 18 shows how sections and paragraphs are used in AsciiRFC, and its rendered form shown in Figure 19.

```
:sectnums:
[toc=exclude]
== Section 1
Para 1

=== Subsection 1.1
Para 1a

:sectnums!:
[toc=default]
=== Subsection 1.2
Para 2

==== Subsection 1.2.1
Para 3
```

Figure 18: AsciiRFC With Sections

```
<section anchor="_section_1" toc="exclude" numbered="true">
  <name>Section 1</name>
  <t>Para 1</t>
  <section anchor="_subsection_1_1" numbered="true">
    <name>Subsection 1.1</name>
    <t>Para 1a</t>
  </section>
  <section anchor="_subsection_1_2" toc="default" numbered="false">
    <name>Subsection 1.2</name>
    <t>Para 2</t>
    <section anchor="_subsection_1_2_1" numbered="false">
      <name>Subsection 1.2.1</name>
      <t>Para 3</t>
    </section>
  </section>
</section>
```

Figure 19: AsciiRFC With Sections (RFC XML v3)

7. Figures

AsciiRFC examples (corresponding to RFC XML Figures), source code Listings, and Literals (preformatted text) are all delimited blocks. Listings and Literals can occur nested within Examples.

An AsciiRFC example with a figure is given in Figure 20, and its rendering in Figure 21.

```
.Figure 1
====
.figure1.txt
....
Figures are only permitted to contain listings
(sourcecode), images (artwork), or literal (artwork)

This is some ASCII Art:

 _____ ___ ____ _      _
|  ___|_ _/ ___| | ___| |_
| |_   | | |  _| |/ _ \ __|
|  _|  | | |_| | |  __/ |_
|_|   |___\____|_|\___|\__|
....

[source,ruby]
----
def listing(node)
  result = []
  if node.parent.context != :example
    result << "<figure>"
  end
end
```

Figure 20: AsciiRFC With A Figure

```
<figure>
  <name>Figure 1</name>
  <artwork type="ascii-art" name="figure1.txt">
    Figures are only permitted to contain listings
    (sourcecode), images (artwork), or literal (artwork)

This is some ASCII Art:

 _____ ___ ____ _      _
|  ___|_ _/ ___| | ___| |_
| |_   | | |  _| |/ _ \ __|
|  _|  | | |_| | |  __/ |_
|_|   |___\____|_|\___|\__|</artwork>
  <sourcecode type="ruby">
    def listing(node)
      result = []
      if node.parent.context != :example
        result &lt;&lt; "&lt;figure&gt;"
      end
    end
  </sourcecode>
</figure>
```

Figure 21: AsciiRFC With A Figure (RFC XML v3)

If an AsciiRFC Listing or Literal occurs outside of an Example (Figure 22), the RFC XML converter will supply the surrounding Figure element (Figure 23).

```
....
This is some ASCII Art:

 _____ ___ ____ _      _
|  ___|_ _/ ___| | ___| |_
| |_   | | |  _| |/ _ \ __|
|  _|  | | |_| | |  __/ |_
|_|   |___\____|_|\___|\__|
....
```

Figure 22: AsciiRFC With ASCII Art Without Figure Wrapping

```
<figure>
  <artwork type="ascii-art">This is some ASCII Art:

 _____ ___ ____ _      _
|  ___|_ _/ ___| | ___| |_
| |_   | | |  _| |/ _ \ __|
|  _|  | | |_| | |  __/ |_
|_|   |___\____|_|\___|\__|</artwork>
</figure>
```

Figure 23: AsciiRFC With ASCII Art Without Figure Wrapping (RFC XML v3)

8. Lists

8.1. Basic Lists

AsciiRFC supports ordered, unordered, and definition lists. Indentation of ordered and unordered lists is indicated by repeating the list item prefix ("*" and "." respectively).

List attributes specify the type of symbol used for ordered lists.

An example of AsciiRFC List is shown in Figure 24 with its rendered version in Figure 25.

```
[loweralpha]
. First
. Second
[upperalpha]
.. Third
.. Fourth
. Fifth
. Sixth
```

Figure 24: AsciiRFC With Lists

```
<ol anchor="id" type="a">
  <li>First</li>
  <li>
    <t>Second</t>
    <ol type="A">
      <li>Third</li>
      <li>Fourth</li>
    </ol>
  </li>
  <li>Fifth</li>
  <li>Sixth</li>
</ol>
```

Figure 25: AsciiRFC With Lists (RFC XML v3)

8.2. List Continuation

A list item by default spans a single paragraph. A following paragraph or other block element can be appended to the current list item by prefixing it with "+" in a separate line.

See the "List Continuation" section in [Asciidoctor-Manual] for more information.

An example of list containuation with text is shown in Figure 26 with its rendered version in Figure 27.

```
Notes::  Note 1.
+
Note 2.
+
Note 3.
```

Figure 26: AsciiRFC List With Text Continuation

```
<dl>
  <dt>Notes</dt>
  <dd>
    <t>Note 1.</t>
    <t>Note 2.</t>
    <t>Note 3.</t>
  </dd>
</dl>
```

Figure 27: AsciiRFC List With Text Continuation (RFC XML v3)

(Multiple paragraphs are not permitted within a list item in RFC XML v2. The RFC XML converter deals with this by converting paragraph breaks into line breaks within a list item.)

List continuations can also be embed to populate a list item with a sequence of blocks as a unit (in an Asciidoctor open block).

An example of list containuation with block is shown in Figure 28 with its rendered version in Figure 29.

```
* List Entry 1
* List Entry 2
+
--
Note 2.

....
Literal
....

Note 3.
--
```

Figure 28: AsciiRFC List With Block Continuation

```
<ul>
  <li>List Entry 1</li>
  <li>
    <t>List Entry 2</t>
    <t>Note 2.</t>
    <figure>
      <artwork type="ascii-art">
        Literal
      </artwork>
    </figure>
    <t>Note 3.</t>
  </li>
</ul>
```

Figure 29: AsciiRFC List With Block Continuation (RFC XML v3)

AsciiDoc, and thus AsciiRFC, considers paragraphs to be the basic level of blocks, and does not permit lists to be nested within them: text after a list is considered to be a new paragraph.

Therefore, markup as shown in Figure 30 cannot be generated via AsciiRFC.

```
<t>
  This is the start of a paragraph.
  <ul>
    <li>List Entry 1</li>
    <li>
      <t>List Entry 2</t>
      <t>Note 2.</t>
    </li>
  </ul>
  And this is the continuation of the paragraph.
</t>
```

Figure 30: This RFC XML v3 Output Cannot Be Generated Using AsciiRFC

9. Blockquotes

Asciidoctor supports blockquotes and quotations of verse; its block quotations permit arbitrary levels of quote nesting. RFC XML v3, and thus AsciiRFC, only supports one level of blockquotes.

Unlike RFC XML v2, RFC XML v3 does not support line breaks outside of tables, so verse quotations are converted to prose in the v3 converter.

An example of using AsciiRFC Blockquotes is given in Figure 31 with its rendered version in Figure 32.

```
[quote,attribution="Monty Python",citetitle="http://example.com"]
____
Dennis: Come and see the violence inherent in the system.
Help! Help! I'm being repressed!

King Arthur: Bloody peasant!

Dennis: Oh, what a giveaway!
* Did you hear that?
* Did you hear that, eh?
* That's what I'm on about!
** Did you see him repressing me?
** You saw him, Didn't you?
____
```

Figure 31: AsciiRFC Blockquote Usage

```
<blockquote quotedfrom="Monty Python" cite="http://example.com">
  <t>Dennis: Come and see the violence inherent in the system.
  Help! Help! I&#8217;m being repressed!</t>
  <t>King Arthur: Bloody peasant!</t>
  <t>Dennis: Oh, what a giveaway!</t>
  <ul>
    <li>Did you hear that?</li>
    <li>Did you hear that, eh?</li>
    <li>
      <t>That&#8217;s what I&#8217;m on about!</t>
      <ul>
        <li>Did you see him repressing me?</li>
        <li>You saw him, Didn&#8217;t you?</li>
      </ul>
    </li>
  </ul>
</blockquote>
```

Figure 32: AsciiRFC Blockquote Usage (RFC XML v3)

10. Notes And Asides

Asciidoctor supports a range of "admonitions", including notes, warnings, and tips. They are indicated by a paragraph prefix (e.g. "WARNING:"), or as a block with an admonition style attribute.

All admonitions are conflated in AsciiRFC, being converted to "note" elements in the document preamble, and "cref" documents in the main document.

This means that all admonitions will therefore not appear in the textual output. A sample of this is shown in Figure 33 with its rendered output in Figure 34.

```
== Section 1
[NOTE,source=GBS]
.Note Title
====
Any admonition inside the body of the text is a comment.
====
```

Figure 33: An AsciiRFC Adminition Block

```
<section anchor="_section_1" numbered="false">
  <name>Section 1</name>
  <t>
    <cref display="true" source="GBS">
      Any admonition inside the body of the text is a comment.
    </cref>
  </t>
</section>
```

Figure 34: An AsciiRFC Adminition Block (RFC XML v3)

With RFC XML v2, note that no inline formatting is permitted for "cref" elements, and is therefore stripped for v2 by the converter.

Because paragraphs in AsciiRFC cannot contain any other blocks, a comment at the end of a paragraph is treated as a new block. In the document converter, any such comments are moved inside the preceding RFC XML paragraph; if the comment is at the start of a section, as in the example above, it is wrapped inside a paragraph.

The RFC XML v3 converter also supports "asides" (Asciidoctor sidebars). A sample is shown in Figure 35 with its rendered output in Figure 36.

```
== Section 1
****
Sidebar

Another sidebar

* This is a list

....
And this is ascii-art
....
****
```

Figure 35: An AsciiRFC Sidebar Block

```
<section anchor="_section_1" numbered="false">
  <name>Section 1</name>
  <aside>
    <t>Sidebar</t>
    <t>Another sidebar</t>
    <ul>
      <li>This is a list</li>
    </ul>
    <figure>
      <artwork type="ascii-art">
        And this is ascii-art
      </artwork>
    </figure>
  </aside>
</section>
```

Figure 36: An AsciiRFC Sidebar Block Rendered As An Aside (RFC XML v3)

Comments given in the AsciiDoc syntax (notated with initial "//") are not intended to be shown in the rendered output, and will not appear in the output as XML comments.

11. Tables

AsciiRFC tables, like RFC XML v3, support distinct table heads, bodies and feet; cells spanning multiple rows and columns; and horizontal alignment. The larger range of table formatting options available in RFC XML v2 is also supported.

A sample of an AsciiRFC table is shown in Figure 37 with its rendered output in Figure 38.

Neither version of RFC XML is as expressive in its table structure as Asciidoctor. RFC XML, for example, does not permit blocks within table cells.

```
   .Table Title
   |===
   |head | head

   h|header cell | body cell
   | | body cell
   2+| colspan of 2
   .2+|rowspan of 2 | cell
   |cell
   ^|centre aligned cell | cell
   <|left aligned cell | cell
   >|right aligned cell | cell

   |foot | foot
   |===
```

Figure 37: An AsciiRFC Table

```
   <table>
     <name>Table Title</name>
     <thead>
       <tr>
         <th align="left">head</th>
         <th align="left">head</th>
       </tr>
     </thead>
     <tbody>
       <tr>
         <th align="left">header cell</th>
         <td align="left">body cell</td>
       </tr>
       <tr>
         <td align="left"></td>
         <td align="left">body cell</td>
       </tr>
       <tr>
         <td colspan="2" align="left">colspan of 2</td>
       </tr>
       <tr>
         <td rowspan="2" align="left">rowspan of 2</td>
         <td align="left">cell</td>
       </tr>
       <tr>
         <td align="left">cell</td>
       </tr>
       <tr>
         <td align="center">centre aligned cell</td>
         <td align="left">cell</td>
```

```
    </tr>
    <tr>
      <td align="left">left aligned cell</td>
      <td align="left">cell</td>
    </tr>
    <tr>
      <td align="right">right aligned cell</td>
      <td align="left">cell</td>
    </tr>
  </tbody>
  <tfoot>
    <tr>
      <td align="left">foot</td>
      <td align="left">foot</td>
    </tr>
  </tfoot>
</table>
```

Figure 38: An AsciiRFC Table (RFC XML v3)

12. Inline Formatting

12.1. Italics, Boldface, Monospace, Subscripts, Superscripts

AsciiRFC supports italics, boldface, monospace, subscripts and superscripts, just like RFC XML v3.

The inline formatting syntax given in Figure 39 produces the RFC XML v3 output given in Figure 40.

```
_Text_ *Text* `Text`  ^Superscript^ ~Subscript~
```

Figure 39: Inline Formatting In AsciiRFC

```
<t><em>Text</em> <strong>Text</strong> <tt>Text</tt>
<sup>Superscript</sup> <sub>Subscript</sub></t>
```

Figure 40: Inline Formatting In AsciiRFC (RFC XML v3)

12.2. Formatting BCP 14 Keywords

RFC XML v3 also supports tagging of BCP14 keywords [RFC2119]; this is done in AsciiRFC either by tagging them with a custom formatting span ("bcp14#must not#"), or by converting BCP14 boldface all-caps words (unless the ":no-rfc-bold-bcp14: false" document attribute is set).

Any spans of BCP14 text delimited by inline formatting delimiters needs to be contained within a single line of text; the Asciidoctor API breaks up formatting spans across line breaks.

This usage is demonstrated in Figure 41 with the rendered output in Figure 42.

```
This [bcp14]#must not# stand

This *MUST NOT* stand
```

Figure 41: BCP14 Keywords In AsciiRFC

```
<t>This <bcp14>MUST NOT</bcp14> stand</t>

<t>This <bcp14>MUST NOT</bcp14> stand</t>
```

Figure 42: BCP14 Keywords In AsciiRFC (RFC XML v3)

12.3. Escaping AsciiRFC Syntax

Formatting delimiters like "*" can be escaped with backslash ("*"); double formatting delimiters, like "**" and "__", need to be escaped with double backslash ("\\**").

Escaping delimiters is not always reliable, and for double delimiters it is preferable to use HTML entities ("&#42;&#42;"), or attribute references (references to the value of attributes set in the document header) as shown in Figure 43.

```
:dblast: **

`{dblast}`
```

Figure 43: Escaping AsciiRFC Syntax Using Attributes

In extreme circumstances (such as quoting AsciiDoc syntax), you may need to resort to altering the substitutions behaviour within a given block of of AsciiDoc; see the "Applying Substitutions" section of [Asciidoctor-Manual].

13. Links

Common URL formats are recognised automatically as hyperlinks in AsciiRFC, inheriting this excellent feature from AsciiDoc, and are rendered as such.

Any hyperlinked text is appended after the hyperlink in square brackets.

An example is given in Figure 44 with its rendered version in Figure 45.

```
http://example.com/[linktext]
```

Figure 44: An AsciiRFC Link

```
<t><eref target="http://example.com/">linktext</eref></t>
```

Figure 45: An AsciiRFC Link (RFC XML v3)

To prevent hyperlinking of a URL, prefix it with a backslash, as shown in Figure 46 with its rendered version in <<source-asciirfc-link-lit-v3>.

```
\http://example.com/[linktext]
```

Figure 46: A Literal AsciiRFC Link

```
<t>http://example.com/[linktext]</t>
```

Figure 47: A Literal AsciiRFC Link (RFC XML v3)

14. Cross-References

14.1. Basic Referencing

Anchors for cross-references are notated as "[[...]]" or "[#...]", and can be inserted on their own line in front of most blocks.

Asciidoctor supports anchors in a much wider range of contexts than is supported than RFC XML v3 (let alone v2); anchors that are not supported for that version of RFC XML are simply ignored by the converter.

Note that anchors in RFC XML are constrained to the format "[A-Za-z_:][[A-Za-z0-9_:.-]*".

Cross-references to anchors are notated as "<<...>>"; cross-references with custom text as "<<reference,text>>".

An example of using cross-references in AsciiRFC is given in Figure 48 with its rendered output in Figure 49.

```
[[cross-reference]]
== Section 1

== Section 2
See <<cross-reference>>.

== Section 3
See <<cross-reference,text>>
```

Figure 48: Setting And Referring To Cross-References In AsciiRFC

```
<section anchor="cross-reference" numbered="false">
  <name>Section 1</name>
</section>
<section anchor="_section_2" numbered="false">
  <name>Section 2</name>
  <t>
    See
    <xref target="cross-reference">
    </xref>.
  </t>
</section>
<section anchor="_section_3" numbered="false">
  <name>Section 3</name>
  <t>
    See
    <xref target="cross-reference">
      text
    </xref>
  </t>
</section>
```

Figure 49: Setting And Referring To Cross-References In AsciiRFC (RFC XML v3)

14.2. Referencing With Attributes

While Asciidoctor natively does not support attributes on cross-references, AsciiRFC works around that by embedding formatting information as templated text within cross-references:

- "format=x: text" populates the "xref@format" attribute

- a section number followed by one of the words "of", "parens", "bare", "text" is treated as a "relref" reference to an external document.

An example of referencing with attributes is given in Figure 50 with its output in Figure 51.

```
== Section 4
See <<cross-reference,format=counter: text>>

== Section 5
See <<cross-reference,format=title>>

See <<cross-reference,1.3 of>>
<<cross-reference,1.4 comma: text>>
<<cross-reference#fragment1,2.5.3 parens>>
<<cross-reference#fragment2,6.2a bare: text>>
```

Figure 50: Cross-References With Attributes In AsciiRFC

```
<section anchor="_section_4" numbered="false">
  <name>Section 4</name>
  <t>
    See
    <xref format="counter" target="cross-reference">
      text
    </xref>
  </t>
</section>
<section anchor="_section_5" numbered="false">
  <name>
    Section 5
  </name>
  <t>
    See
    <xref format="title" target="cross-reference" />
  </t>
  <t>
    See
    <relref section="1.3" displayformat="of"
     target="cross-reference" />
    <relref section="1.4" displayformat="comma"
     target="cross-reference">
      text
    </relref>
    <relref relative="fragment1" section="2.5.3"
     displayformat="parens" target="cross-reference" />
    <relref relative="fragment2" section="6.2a"
     displayformat="bare" target="cross-reference">
      text
    </relref>
  </t>
</section>
```

Figure 51: Cross-References With Attributes In AsciiRFC (RFC XML v3)

15. Inclusions

AsciiRFC inherits the Asciidoctor "include" directive [Asciidoctor-Manual] to include external files in a master AsciiRFC document.

This directive is capable of sophisticated document merging, including adjusting the heading levels of the included text, selecting text within specified tags or line numbers to be included, and adjusting the indentation of code snippets in merged text.

Its basic syntax is given in Figure 52.

```
include::path[
  leveloffset=_offset_,
  lines=_ranges_,
  tag(s)=_name(s)_,
  indent=_depth_
]
```

Figure 52: Inclusions In AsciiRFC

If a file is included in an AsciiRFC document, ensure it ends with a blank line. An inclusion that results in its final block not being delimited with a blank line from what follows can lead to unpredictable results.

16. Encoding and Entities

XML accepts the full range of characters in the world's languages through UTF-8 character encoding, and one of the motivations for the move from plain text to RFC XML has been to allow non-ASCII characters to be included in RFCs.

However, current RFC XML v2 tools still do not support UTF-8, and alternative tooling support for UTF-8 also remains patchy. Out of an abundance of caution, the RFC XML converter uses US-ASCII for its character encoding, and renders any non-ASCII characters as HTML entities.

AsciiRFC accepts HTML entities as input even though they are not part of the XML specification. HTML entities such as "&nbsp;" feature in examples of RFC XML provided by the IETF. In order to prevent dependence of the XML output from extraneous entity definitions, any such entities are rendered in the XML as decimal character entities.

An example of how AsciiRFC renders non-ASCII UTF-8 characters are given in Figure 53 with the output in Figure 54.

```
&#1069;&#1090;&#1086;
&#1056;&#1091;&#1089;&#1089;&#1082;&#1080;&#1081;
&#1071;&#1079;&#1099;&#1082;.
&mdash; This is not George&apos;s.&#x2020;
```

Figure 53: UTF-8 Characters In AsciiRFC

```
<t>&#1069;&#1090;&#1086;
&#1056;&#1091;&#1089;&#1089;&#1082;&#1080;&#1081;
&#1071;&#1079;&#1099;&#1082;.
&#8212; This is not George's.&#8224;</t>
```

Figure 54: UTF-8 Characters In AsciiRFC Rendered As RFC XML v3

17. Bibliography

The simple encoding of bibliography syntax provided by AsciiDoc (and Asciidoctor) is inadequate for the complexity of bibliographic markup required by RFC XML.

RFC documents overwhelmingly cite other RFC documents, and canonical RFC XML bibliographic entries are available at [IETF-BibXML]; so it would be inefficient to encode those entries in AsciiRFC, only to have them converted back to RFC XML.

The converter provides two means of incorporating bibliographies into RFC documents authored in AsciiRFC:

- using raw RFC XML; and

- assembling bibliographies in preprocessing.

In either case, the RFC XML needs to be well-formed; missing closing tags can lead to erratic behaviour in the converter.

17.1. Using Raw RFC XML

In the first method, bibliographic citations are handled like all other AsciiRFC cross-references. The bibliographic entries for normative and informative references are given in the AsciiRFC as passthrough blocks, which contain the raw RFC XML for all references; document conversion leaves the raw RFC XML in place.

This approach requires authors to maintain the normative and informative bibliographies within the document, to update them as citations are added and removed, and to sort them manually.

For example, the AsciiRFC in Figure 55 will generate the corresponding RFC XML in Figure 56.

```
Some datagram padding may be needed.<<RFC7253>>

[bibliography]
== Normative References
++++
<reference anchor='RFC7253'
  target='https://tools.ietf.org/html/rfc7253'>
  <front>
    <title>Guidelines for Writing an IANA Considerations
      Section in RFCs</title>
    <author initials="T." surname="Krovetz">
      <organization>Sacramento State</organization>
    </author>
    <author initials="P." surname="Rogaway">
      <organization>UC Davis</organization>
    </author>
    <date month='May' year='2014'/>
  </front>
  <seriesInfo name="RFC" value="7253"/>
</reference>
++++
```

Figure 55: AsciiRFC Inline Bibliography

```
  <t>Some datagram padding may be needed <xref target="RFC7253"/></t>
</middle>

<back>
<references anchor="_references">
  <name>Normative References</name>
  <reference anchor='RFC7253'
    target='https://tools.ietf.org/html/rfc7253'>
    <front>
      <title>Guidelines for Writing an IANA Considerations
        Section in RFCs</title> <author initials="T." surname="Krovetz">
        <organization>Sacramento State</organization>
      </author>
      <author initials="P." surname="Rogaway">
        <organization>UC Davis</organization>
      </author>
      <date month='May' year='2014'/>
    </front>
    <seriesInfo name="RFC" value="7253"/>
  </reference>
</references>
</back>
```

Figure 56: AsciiRFC Inline Bibliography Rendered As RFC XML v3

17.2. Preprocessing Using <spanx style="verb">asciidoctor-bibliography</spanx>

The alternative method is to use a preprocessing tool, [asciidoctor-bibliography], to import citations into the AsciiRFC document from an external file of references.

The references file consists of RFC XML reference entries, and still needs to be managed manually; however the bibliographies are assembled from that file, sorted, and inserted into the normative and informative references in preprocessing. Citations in the document itself are given as macros to be interpreted by the preprocessor; this allows them to be split into normative and informative references. (The MMark tool likewise splits reference citations into normative and informative.)

Integration with the "asciidoc-bibliography" gem proceeds as follows:

1. Create an RFC XML references file, consisting of a "<references>" element with individual "<reference>" elements inserted, as would be done for the informative and normative references normally. The references file will contain all possible references to be used in the file; the bibliography gem will select which references have actually been cited in the document.

   A. Rather than hand crafting RFC XML references for RFC documents, you should download them from an authoritative source; e.g. "http://xml.resource.org/public/rfc/bibxml/reference.RFC.2119.xml"

   B. Unlike the case for RFC XML documents created manually, the references file does not recognise XML entities and will not attempt to download them during processing. Any references to "http://xml.resource.org/public/rfc/bibxml/" will need to be downloaded and inserted into the references file.

   C. The RFC XML in the references file will need to be appropriate to the version of RFC XML used in the main document, as usual. Note that RFC XML v2 references are forward compatible with v3; v3 contains a couple of additional elements.

2. Add to the main document header attributes referencing the references file (":bibliography-database:"), and the bibliography style (":bibliography-style: rfc-v3").

3. References to a normative reference are inserted with the macro "cite:norm[id]" instead of "<<id>>", where "id" is the anchor of the reference.

4. References to an infomrative reference are inserted with the macro "cite:info[id]" instead of "<<id>>", where "id" is the anchor of the reference.

5. Formatted crossreferences and "relref" crossreferences are entered by inserting the expected raw XML in the "text" attribute. Do not use the "{cite}" interpolation of the citation. For example:

   * "<<id,words>>" = "cite:norm[id, text="<xref target='id'>words</xref>"]"

   * "<<id,format=counter: words>>" (processed as a formatted cross-reference) = "cite:norm[id, text="<xref format='counter' target='id'>words</xref>"]"

   * "<<id,2.4 comma: words>>" (processed as relref) = "cite:norm[id, text="<relref displayFormat='comma' section='2.4' target='id'}/>"]"

   * "<<id#section2_4,2.4 comma: words>>" (processed as relref with a cross-document internal reference) = "cite:norm[id, text="<relref relative='section2_4' displayFormat='comma' section='2.4' target='id'/>"]"

6. Normative and Informative References are inserted in the document through a macro, which occurs where the RFC XML references would be inserted, as shown in Figure 57.

```
[bibliography]
== Normative References

++++
bibliography::norm[]
++++

[bibliography]
== Informative References

++++
bibliography::info[]
++++
```

Figure 57: Using asciidoctor-bibliography For Bibliography Preprocessing

18. RFC XML features not supported in Asciidoctor

The following features of RFC XML v3 [RFC7991] and v2 [RFC7749] are not supported by the AsciiRFC converter, and would need to be adjusted manually after RFC XML is generated:

RFC XML element	RFC XML v3	RFC XML v2
"front/boilerplate"	Not added by the converter	Not added by the converter
"iref@primary"	N	N
"reference" (and all children)	As Raw XML	As Raw XML
"table/preamble"	Deprecated	N
"table/postamble"	Deprecated	N
"artwork@width"	Only on images	Only on images
"artwork@height"	Only on images	Only on images

19. Authoring

To author an AsciiRFC document, you should first familiarise yourself with the [Asciidoctor-Manual].

The [asciidoctor-rfc] Ruby gem source code distribution also has samples of individual RFC XML features in v2 and v3, and examples of self-standing AsciiRFC documents, along with their RFC XML renderings. (This includes round-tripped RFC XML documents.)

19.1. Using the <spanx style="verb">rfc-in-asciidoc-template</spanx>

In addition, you can clone the sample "rfc-in-asciidoc-template" repository as a template, and populate it for your AsciiRFC document using the steps shown in Figure 58.

```
   $ git clone https://github.com/riboseinc/rfc-in-asciidoc-template
```

Figure 58: Cloning The AsciiRFC Document Template

19.2. Installing AsciiRFC Backend Processors

Converting your AsciiRFC to RFC XML is a simple as installing Asciidoctor (see "Installation" at [Asciidoctor]) and the "asciidoctor-rfc" gem in Ruby (through RubyGems), then running the asciidoctor executable on the document, specifying the asciidoctor-rfc gem as a library.

The necessary steps are shown in Figure 59.

```
$ gem install asciidoctor-rfc
$ asciidoctor -b rfc3 -r 'asciidoctor-rfc' foo.adoc  # RFC XML v3 output
$ asciidoctor -b rfc2 -r 'asciidoctor-rfc' foo.adoc  # RFC XML v2 output
```

Figure 59: Installing The AsciiRFC Backend Processors

19.3. Iterating AsciiRFC Content

As you author AsciiRFC content, you should iterate through running the Asciidoctor conversion frequently, to ensure that you are still generating valid XML through your markup. The converter makes an effort to ensure that its XML output is valid, and it issues warnings about likely issues; it also validates its own XML output against the RFC XML schema (of the corresponding version), and reports errors in the XML output in the format shown in Figure 60.

```
   V3 RELAXNG Validation: 12:0: ERROR: Invalid attribute
     sortRefs for element rfc
```

Figure 60: Sample Validation Error Message From AsciiRFC

Note that validation against the RELAXNG RFC XML schema includes confirming the referential integrity of all cross-references in the document.

It may be necessary to intervene in the XML output generated by the converter, either because the block model of AsciiRFC does not conform with the intended RFC XML (e.g. lists embedded in

paragraphs), or because RFC XML features are required that are not supported within AsciiRFC.

20. Security Considerations

- Ensure your AsciiRFC generator comes from a geniune and trustworthy source. This protects your own machine and also prevents injection of malicious content in your resulting document.

- An AsciiRFC generator may cause errors in textual rendering or link generation that may lead to security issues.

- Creating cross-references (and also bibliographic references) to external documents may pose risks since the specified external location may become controlled by a malicious party.

21. IANA Considerations

This document does not require any action by IANA.

22. Examples

22.1. Example 1: AsciiRFC

```
= The Holy Hand Grenade of Antioch
Arthur Pendragon <arthur@camelot.gov.uk>
:doctype: internet-draft
:name: draft-iab-holy-hand-grenade-antioch-00
:status: informational
:ipr: trust200902
:toc-include: true
:forename_initials: A.
:organization: Camelot
:revdate: 932-06-23
:area: General
:workgroup: Internet Architecture Board
:smart-quotes: false

[abstract]
The Killer Rabbit of Caerbannog is a fictional character in the Monty
Python film _Monty Python and the Holy Grail_. The scene in _Holy Grail_
was written by Graham Chapman and John Cleese. The rabbit is the
antagonist in a major set piece battle, and makes a similar appearance
in _Spamalot_, a musical inspired by the movie.

== Terminology
In this document, the key words *MUST*, *MUST NOT*, *REQUIRED*,
```

```
*SHALL*, *SHALL NOT*, *SHOULD*, *SHOULD NOT*, *RECOMMENDED*, *MAY*, and
*OPTIONAL* are to be interpreted as described in BCP 14, <<RFC2119>>.

== In The Film

The Cave of Caerbannog (_caer bannog_ being Welsh for "turreted castle",
thus making its title a pun on the English dish "Welsh rabbit") is the
home of the Legendary Black Beast of Arrrghhh (named for the last
utterance of anyone who ever saw it). This is guarded by a monster
which is initially unknown. King Arthur and his knights are led to the
cave by Tim the Enchanter and find that they must face its guardian
beast.

=== Holy Hand Grenade of Antioch

The Holy Hand Grenade of Antioch is a visual satire of the Sovereign's
Orb of the United Kingdom, and may refer to the mythical Holy Spear of
Antioch. The Holy Hand Grenade is described as one of the
"sacred relics" carried by Brother Maynard. Despite its ornate
appearance and long-winded instructions, it functions much the same
as any other hand grenade.

[bibliography]
== Normative References
++++
<reference anchor="RFC2119"
    target="https://www.rfc-editor.org/info/rfc2119">
  <front>
    <title>Key words for use in RFCs to Indicate
      Requirement Levels</title>
    <author initials="S." surname="Bradner" fullname="S. Bradner">
      <organization/>
    </author>
    <date year="1997" month="March"/>
  </front>
  <seriesInfo name="BCP" value="14"/>
  <seriesInfo name="RFC" value="2119"/>
  <seriesInfo name="DOI" value="10.17487/RFC2119"/>
</reference>
++++

[appendix]
== Cultural Impact

The rabbit is now used as a metaphor for something ostensibly harmless
which is, in fact, deadly. Such hidden but real risks may even arise
from similarly cuddly animals.
```

22.2. Example 1: RFC XML v3

```
<?xml version="1.0" encoding="US-ASCII"?>
<!DOCTYPE rfc SYSTEM "rfc2629.dtd">
<?rfc strict="yes"?>
<?rfc compact="yes"?>
<?rfc subcompact="no"?>
<?rfc toc="yes"?>
<?rfc tocdepth="4"?>
<?rfc symrefs="yes"?>
<?rfc sortrefs="yes"?>
<rfc xmlns:xi="http://www.w3.org/2001/XInclude" ipr="trust200902"
    tocInclude="true" submissionType="IETF"
    prepTime="2017-12-01T10:19:27Z" version="3">
  <front>
    <title>The Holy Hand Grenade of Antioch</title>
    <seriesInfo name="Internet-Draft" status="informational"
      stream="IETF" value="draft-iab-holy-hand-grenade-antioch-00" />
    <author fullname="Arthur Pendragon" surname="Pendragon"
        initials="A.">
      <organization>Camelot</organization>
      <address>
        <email>arthur@camelot.gov.uk</email>
      </address>
    </author>
    <date day="23" month="June" year="932" />
    <area>General</area>
    <workgroup>Internet Architecture Board</workgroup>
    <abstract>
      <t>The Killer Rabbit of Caerbannog is a fictional character in
        the Monty Python film  <em>Monty Python and the Holy
        Grail</em>. The scene in <em>Holy Grail</em> was written by
        Graham Chapman and John Cleese. The rabbit is the antagonist
        in a major set piece battle, and makes a similar appearance
        in <em>Spamalot</em>, a musical inspired by the movie.</t>
    </abstract>
  </front>
  <middle>
    <section anchor="_terminology" numbered="false">
      <name>Terminology</name>
      <t>In this document, the key words
        <bcp14>MUST</bcp14>,
        <bcp14>MUST NOT</bcp14>,
        <bcp14>REQUIRED</bcp14>,
        <bcp14>SHALL</bcp14>,
        <bcp14>SHALL NOT</bcp14>,
        <bcp14>SHOULD</bcp14>,
        <bcp14>SHOULD NOT</bcp14>,
```

```
         <bcp14>RECOMMENDED</bcp14>,
         <bcp14>MAY</bcp14>, and
         <bcp14>OPTIONAL</bcp14> are to be interpreted as described in
         BCP 14, <xref target="RFC2119" />.</t>
     </section>
     <section anchor="_in_the_film" numbered="false">
       <name>In The Film</name>
       <t>The Cave of Caerbannog (<em>caer bannog</em> being Welsh for
         "turreted castle", thus making its title a pun on the English
         dish "Welsh rabbit") is the home of the Legendary Black Beast
         of Arrrghhh (named for the last utterance of anyone who ever
         saw it). This is guarded by a monster which is initially
         unknown. King Arthur and his knights are led to the cave by
         Tim the Enchanter and find that they must face its guardian
         beast.</t>
       <section anchor="_holy_hand_grenade_of_antioch" numbered="false">
         <name>Holy Hand Grenade of Antioch</name>
         <t>The Holy Hand Grenade of Antioch is a visual satire of the
           Sovereign's Orb of the United Kingdom, and may refer to the
           mythical Holy Spear of Antioch. The Holy Hand Grenade is
           described as one of the "sacred relics" carried by Brother
           Maynard. Despite its ornate appearance and long-winded
           instructions, it functions much the same as any other hand
           grenade.</t>
       </section>
     </section>
   </middle>
   <back>
     <references anchor="_normative_references">
       <name>Normative References</name>
       <xi:include
 href="http://xml2rfc.ietf.org/public/rfc/bibxml/reference.RFC.2119.xml"
 parse="text" />
     </references>
     <section anchor="_cultural_impact" numbered="false">
       <name>Cultural Impact</name>
       <t>The rabbit is now used as a metaphor for something ostensibly
         harmless which is, in fact, deadly. Such hidden but real risks
         may even arise from similarly cuddly animals.</t>
     </section>
   </back>
 </rfc>
```

22.3. Example 2: AsciiRFC

```
= The Holy Hand Grenade of Antioch
:doctype: internet-draft
:abbrev: Hand Grenade of Antioch
```

```
:submission-type: independent
:name: draft-iab-holy-hand-grenade-antioch-01
:status: informational
:consensus: false
:ipr: trust200902
:toc-include: true
:fullname: Arthur son of Uther Pendragon
:forename_initials: A.
:lastname: Pendragon
:email: arthur@camelot.gov.uk
:forename_initials: A.
:organization: Camelot
:uri: http://camelot.gov.uk
:street: Palace\ Camel Lot 1
:city: Camelot
:country: England
:fullname_2: Patsy
:lastname_2: Patsy
:role_2: editor
:email_2: patsy@camelot.gov.uk
:organization_2: Camelot
:postal-line_2: Camel Lot 9\ Camelot\ England
:revdate: 932-06-23
:area: General, Operations and Management
:workgroup: Internet Architecture Board
:keyword: rabbits, grenades
:smart-quotes: false
:obsoletes: 10, 20
:updates: 2010
:sort-refs: true
:comments: yes
:notedraftinprogress: yes
:link: https://en.wikipedia.org/wiki/Rabbit_of_Caerbannog
  convertedFrom,  http://questionthekillerrabbit.tumblr.com preview

[abstract]
The Killer Rabbit of Caerbannog is a fictional character in the Monty
Python film _Monty Python and the Holy Grail_. The scene in _Holy Grail_
was written by Graham Chapman and John Cleese. The rabbit is the
antagonist in a major set piece battle, and makes a similar appearance
in _Spamalot_, a musical inspired by the movie. See also
<<RFC2635,1 of What is Spam*?>>

[NOTE,remove-in-rfc=false]
.Spamalot
The iconic status of this scene was important in establishing
the viability of the musical.
```

```
[toc=exclude]
:sectnums!:
== Terminology
In this document, the key words *MUST*, *MUST NOT*, *REQUIRED*,
*SHALL*, *SHALL NOT*, *SHOULD*, *SHOULD NOT*, *RECOMMENDED*, *MAY*, and
*OPTIONAL* are to be interpreted as described in BCP 14, <<RFC2119>>.

:sectnums:
== In The Film
The Cave of Caerbannog (_caer bannog_ being Welsh for "turreted
castle", thus making its title a pun on the English dish "Welsh
rabbit") is the home of the Legendary Black Beast of Arrrghhh
(((Killer Rabbit of Caerbannog)))
(named for the last utterance of anyone who ever saw it). This is
guarded by a monster which is initially unknown. ((King Arthur)) and
his knights are led to the cave by ((Tim the Enchanter)) and find that
they must face its guardian beast.

****
The rabbit scene was shot outside the Tomnadashan mine, a cave 4 miles
(6.5 km) from the Perthshire village of Killin. For the 25th
anniversary DVD, Michael Palin and Terry Jones returned to be
interviewed in front of the cave but they could not remember the
location.
****

* Tim verbally paints a picture of
a terrible monster with "nasty, big, pointy teeth!", so terrifying
that Sir Robin soils his armour at the mere description.
(((Sir Robin, soiling armour)))
* When the
guardian appears to be an innocuous white rabbit
(<<killer_bunny,See depiction>>:
http://ascii.co.uk/art/rabbit[RABBIT - ASCII ART]), surrounded
by the bones of the fallen, Arthur and his knights no longer take it
seriously.
** Ignoring Tim's warnings ("a vicious streak a mile wide!"),
King Arthur
orders Bors to chop its head off.
[upperalpha,group=Victims]
... Bors confidently approaches it,
sword drawn, and is immediately decapitated by the rabbit biting
clean through his neck, to the sound of a can opener.
** Despite their
initial shock, Sir Robin soiling his armor again, and Tim's loud
scoffing, the knights attack in force.
[upperalpha,group=Victims]
... But the rabbit injures several
```

```
of the knights and kills Gawain and Ector with ease. The knights
themselves have no hope of killing or injuring the rabbit.
** Arthur
panics and shouts for the knights to retreat ("Run away!").
* Knowing
they cannot risk attacking again, they try to find another way to
defeat the beast.
* The Holy Hand Grenade of Antioch is ultimately
used to kill it and allow the quest to proceed.

[NOTE,display=false,source=Lancelot]
.Tip for the Bridge scene
What is Lancelot's favourite colour? Will come in handy later.

[[killer_bunny]]
.Figure 1
====
[alt=Killer Bunny]
....

           /\ /|
          |||| |
           \ | \
       _ _ /  @ @
     /    \   =>X<=
   /|      |   /
   \|     /__| |
     \_____\ \__\


unknown
....
====

.Dramatis Personae
[grid=all]
|===
|Actor |Role

|Graham Chapman >|King Arthur
|John Cleese >|Tim the Enchanter
.2+|Eric Idle >|Sir Robin
>|Brother Maynard
|Terry Gilliam >|Sir Bors
|Michael Palin >|The Lector
|===

=== Holy Hand Grenade of Antioch
```

```
[[sovereign_orb]]
.Figure 2
====
.Sovereign's Orb
[link=https://en.wikipedia.org/wiki/File:British_Sovereigns_Orb.jpg,
  align=right]
image::https://en.wikipedia.org/wiki/File:British_Sovereigns_Orb.jpg
  [Orb,124,135]
====

The Holy Hand Grenade of Antioch is a visual satire of the Sovereign's
Orb of the United Kingdom, Figure <<sovereign_orb,format=counter>>,
and may refer to the mythical Holy Spear of Antioch. The Holy Hand
Grenade is described as one of the "sacred relics" carried by Brother
Maynard. Despite its ornate appearance and long-winded instructions,
it functions much the same as any other hand grenade. At King Arthur's
prompting, instructions for its use are read aloud from the fictitious
_Book of Armaments_, Chapter 2, verses 9-21.

NOTE: Verses parodying the King James Bible and the Athanasian Creed.

[keep-with-previous=true]
[quote,Book of Armaments 2:9-21,
  https://genius.com/Monty-python-holy-hand-grenade-of-antioch-lyrics]
And Saint Attila raised the hand grenade up on high, saying,
"O *LORD*, bless this Thy hand grenade that with it Thou
[bcp14]#mayest# blow Thine enemies to tiny bits, in Thy mercy." And
the *LORD* did grin and
the people did feast upon the lambs and sloths and carp and anchovies
and orangutans and breakfast cereals, and fruit bats and large chu...
[At this point, the friar is urged by ((Brother Maynard)) to
"skip a bit, brother"]... And the *LORD* spake, saying, "First
[bcp14]#shalt# thou take out the Holy Pin, then [bcp14]#shalt# thou
count to three, no more, no less. Three
[bcp14]#shall# be the number thou [bcp14]#shalt# count, and the number
of the counting [bcp14]#shall# be three. Four [bcp14]#shalt# thou not
count, neither count thou two,
excepting that thou then proceed to three. Five is right out. Once
the number three, being the third number, be reached, then lobbest
thou thy Holy Hand Grenade of Antioch towards thy foe, who being
naughty in My sight, [bcp14]#shall# snuff it."

=== Code Example

.Sample Python program
[source,python,align=center]
----
ready = True
```

```
if ready:
    print("Hello World!")
----


[bibliography]
== Normative References
++++
<reference anchor="RFC2119"
    target="https://www.rfc-editor.org/info/rfc2119">
  <front>
    <title>Key words for use in RFCs to Indicate
      Requirement Levels</title>
    <author initials="S." surname="Bradner" fullname="S. Bradner">
      <organization/>
    </author>
    <date year="1997" month="March"/>
  </front>
  <seriesInfo name="BCP" value="14"/>
  <seriesInfo name="RFC" value="2119"/>
  <seriesInfo name="DOI" value="10.17487/RFC2119"/>
</reference>
++++

[bibliography]
== Informative References
++++
<reference anchor="RFC2635"
    target="https://www.rfc-editor.org/info/rfc2635">
  <front>
    <title>DON'T SPEW A Set of Guidelines for Mass Unsolicited
    Mailings and Postings (spam*)</title>
    <author initials="S." surname="Hambridge" fullname="S. Hambridge">
      <organization />
    </author>
    <author initials="A." surname="Lunde" fullname="A. Lunde">
      <organization />
    </author>
    <date year="1999" month="June" />
  </front>
  <seriesInfo name="FYI" value="35" />
  <seriesInfo name="RFC" value="2635" />
  <seriesInfo name="DOI" value="10.17487/RFC2635" />
</reference>
++++
```

22.4. Example 2: RFC XML v3

```
<?xml version="1.0" encoding="US-ASCII"?>
<!DOCTYPE rfc SYSTEM "rfc2629.dtd">
<?rfc comments="yes"?>
<?rfc notedraftinprogress="yes"?>
<?rfc strict="yes"?>
<?rfc compact="yes"?>
<?rfc subcompact="no"?>
<?rfc toc="yes"?>
<?rfc tocdepth="4"?>
<?rfc symrefs="yes"?>
<?rfc sortrefs="true"?>
<rfc xmlns:xi="http://www.w3.org/2001/XInclude" ipr="trust200902"
    obsoletes="10, 20" updates="2010" sortRefs="true" tocInclude="true"
    submissionType="independent" prepTime="2017-12-01T13:28:00Z"
    version="3">
  <link href="https://en.wikipedia.org/wiki/Rabbit_of_Caerbannog"
    rel="convertedFrom" />
  <link href="http://questionthekillerrabbit.tumblr.com"
    rel="preview" />
  <front>
    <title abbrev="Hand Grenade of Antioch">The Holy Hand Grenade
      of Antioch</title>
    <seriesInfo name="Internet-Draft" status="informational"
      stream="independent"
      value="draft-iab-holy-hand-grenade-antioch-01" />
    <author fullname="Arthur son of Uther Pendragon"
        surname="Pendragon" initials="A.">
      <organization>Camelot</organization>
      <address>
        <postal>
          <street>Palace</street>
          <street>Camel Lot 1</street>
          <city>Camelot</city>
          <country>England</country>
        </postal>
        <email>arthur@camelot.gov.uk</email>
        <uri>http://camelot.gov.uk</uri>
      </address>
    </author>
    <author fullname="Patsy" surname="Patsy" role="editor">
      <organization>Camelot</organization>
      <address>
        <postal>
          <postalLine>Camel Lot 9</postalLine>
          <postalLine>Camelot</postalLine>
          <postalLine>England</postalLine>
```

```
        </postal>
        <email>patsy@camelot.gov.uk</email>
      </address>
    </author>
    <date day="23" month="June" year="2832" />
    <area>General</area>
    <area>Operations and Management</area>
    <workgroup>Internet Architecture Board</workgroup>
    <keyword>rabbits</keyword>
    <keyword>grenades</keyword>
    <abstract>
      <t>The Killer Rabbit of Caerbannog is a fictional character in
        the Monty Python film
        <em>Monty Python and the Holy Grail</em>. The scene in
        <em>Holy Grail</em> was written by Graham Chapman and John
        Cleese. The rabbit is the antagonist in a major set piece
        battle, and makes a similar appearance in
        <em>Spamalot</em>, a musical inspired by the movie. See also
        <relref section="1" displayFormat="of" target="RFC2635">What
        is Spam*?</relref>
      </t>
    </abstract>
    <note removeInRFC="false">
      <name>Spamalot</name>
      <t>The iconic status of this scene was important in establishing
        the viability of the musical.</t>
    </note>
  </front>
  <middle>
    <section anchor="_terminology" toc="exclude" numbered="false">
      <name>Terminology</name>
      <t>In this document, the key words
        <bcp14>MUST</bcp14>,
        <bcp14>MUST NOT</bcp14>,
        <bcp14>REQUIRED</bcp14>,
        <bcp14>SHALL</bcp14>,
        <bcp14>SHALL NOT</bcp14>,
        <bcp14>SHOULD</bcp14>,
        <bcp14>SHOULD NOT</bcp14>,
        <bcp14>RECOMMENDED</bcp14>,
        <bcp14>MAY</bcp14>, and
        <bcp14>OPTIONAL</bcp14> are to be interpreted as described in
        BCP 14, <xref target="RFC2119" />.</t>
    </section>
    <section anchor="_in_the_film" numbered="true">
      <name>In The Film</name>
      <t>The Cave of Caerbannog (<em>caer bannog</em> being Welsh for
        "turreted castle", thus making its title a pun on the English
```

```
  dish "Welsh rabbit") is the home of the Legendary Black Beast
  of Arrrghhh <iref item="Killer Rabbit of Caerbannog" />
  (named for the last utterance of anyone who ever saw it). This
  is guarded by a monster which is initially unknown. King
  Arthur <iref item="King Arthur" /> and his knights are led to
  the cave by Tim the Enchanter
  <iref item="Tim the Enchanter" /> and find that they must
  face its guardian beast.</t>
<aside>
  <t>The rabbit scene was shot outside the Tomnadashan mine, a
  cave 4 miles (6.5 km) from the Perthshire village of Killin.
  For the 25th anniversary DVD, Michael Palin and Terry Jones
  returned to be interviewed in front of the cave but they could
  not remember the location.</t>
</aside>
<ul>
  <li>Tim verbally paints a picture of a terrible monster with
    "nasty, big, pointy teeth!", so terrifying that Sir Robin
    soils his armour at the mere description.
    <iref item="Sir Robin" subitem="soiling armour" />
  </li>
  <li>
    <t>When the guardian appears to be an innocuous white rabbit
      (<xref target="killer_bunny">See depiction</xref>:
      <eref target="http://ascii.co.uk/art/rabbit">RABBIT -
      ASCII ART</eref>), surrounded by the bones of the fallen,
      Arthur and his knights no longer take it seriously.</t>
    <ul>
      <li>
        <t>Ignoring Tim's warnings ("a vicious streak a mile
        wide!"), King Arthur orders Bors to chop its head off.</t>
        <ol group="Victims" type="A">
          <li>Bors confidently approaches it, sword drawn, and is
            immediately decapitated by the rabbit biting clean
            through his neck, to the sound of a can opener.</li>
        </ol>
      </li>
      <li>
        <t>Despite their initial shock, Sir Robin soiling his
          armor again, and Tim's loud scoffing, the knights attack
          in force.</t>
        <ol group="Victims" type="A">
          <li>But the rabbit injures several of the knights and
            kills Gawain and Ector with ease. The knights
            themselves have no hope of killing or injuring the
            rabbit.</li>
        </ol>
      </li>
```

```
            <li>Arthur panics and shouts for the knights to retreat
              ("Run away!").</li>
          </ul>
        </li>
        <li>Knowing they cannot risk attacking again, they try to find
          another way to defeat the beast.</li>
        <li>The Holy Hand Grenade of Antioch is ultimately used to
          kill it and allow the quest to proceed.</li>
      </ul>
      <t>
        <cref display="false" source="Lancelot">What is Lancelot's
          favourite colour? Will come in handy later.</cref>
      </t>
      <figure anchor="killer_bunny">
        <name>Figure 1</name>
        <artwork type="ascii-art" alt="Killer Bunny">

           /\ /|
          |||| |
           \ | \
       _ _ /  @ @
     /    \   =>X<=
   /|      |   /
   \|     /__| |
     \_____\ \__\


unknown
        </artwork>
      </figure>
      <table>
        <name>Dramatis Personae</name>
        <thead>
          <tr>
            <th align="left">Actor</th>
            <th align="left">Role</th>
          </tr>
        </thead>
        <tbody>
          <tr>
            <td align="left">Graham Chapman</td>
            <td align="right">King Arthur</td>
          </tr>
          <tr>
            <td align="left">John Cleese</td>
            <td align="right">Tim the Enchanter</td>
          </tr>
          <tr>
```

```
            <td rowspan="2" align="left">Eric Idle</td>
            <td align="right">Sir Robin</td>
          </tr>
          <tr>
            <td align="right">Brother Maynard</td>
          </tr>
          <tr>
            <td align="left">Terry Gilliam</td>
            <td align="right">Sir Bors</td>
          </tr>
          <tr>
            <td align="left">Michael Palin</td>
            <td align="right">The Lector</td>
          </tr>
        </tbody>
      </table>
      <section anchor="_holy_hand_grenade_of_antioch" numbered="true">
        <name>Holy Hand Grenade of Antioch</name>
        <figure anchor="sovereign_orb">
          <name>Figure 2</name>
          <artwork align="right" alt="Orb" height="135"
            name="Sovereign's Orb"
src="https://en.wikipedia.org/wiki/File:British_Sovereigns_Orb.jpg"
            type="binary-art" width="124" />
        </figure>
        <t>The Holy Hand Grenade of Antioch is a visual satire of the
          Sovereign's Orb of the United Kingdom, Figure
          <xref format="counter" target="sovereign_orb" />, and may
          refer to the mythical Holy Spear of Antioch. The Holy Hand
          Grenade is described as one of the "sacred relics" carried
          by Brother Maynard. Despite its ornate appearance and
          long-winded instructions, it functions much the same as any
          other hand grenade. At King Arthur's prompting, instructions
          for its use are read aloud from the fictitious
          <em>Book of Armaments</em>, Chapter 2, verses 9-21.
          <cref>Verses parodying the King James Bible and the
          Athanasian Creed.</cref>
        </t>
        <blockquote quotedFrom="Book of Armaments 2:9-21"
          cite="https://genius.com/Monty-python-holy-
          hand-grenade-of-antioch-lyrics">And
          Saint Attila raised the hand grenade up on high, saying,
          "O <strong>LORD</strong>, bless this Thy hand grenade that
          with it Thou <bcp14>MAYEST</bcp14> blow Thine enemies to
          tiny bits,
          in Thy mercy." And the <strong>LORD</strong> did grin and
          the people did feast upon the lambs and sloths and carp and
          anchovies and orangutans and breakfast cereals, and fruit
```

```
          bats and large
          chu&#8230;&#8203; [At this point, the friar is urged by
          Brother Maynard <iref item="Brother Maynard"/> to "skip a
          bit, brother"]&#8230;&#8203; And the <strong>LORD</strong>
          spake, saying, "First <bcp14>SHALT</bcp14>
          thou take out the Holy Pin, then <bcp14>SHALT</bcp14> thou
          count to three, no more, no less. Three
          <bcp14>SHALL</bcp14> be the number thou
          <bcp14>SHALT</bcp14> count, and the number of the counting
          <bcp14>SHALL</bcp14> be three. Four <bcp14>SHALT</bcp14>
          thou not count, neither count thou two, excepting that thou
          then proceed to three. Five is right out. Once
          the number three, being the third number, be reached, then
          lobbest thou thy Holy Hand Grenade of Antioch towards thy
          foe, who being naughty in My sight,
          <bcp14>SHALL</bcp14> snuff it."</blockquote>
      </section>
      <section anchor="_code_example" numbered="true">
        <name>Code Example</name>
        <figure>
          <sourcecode name="Sample Python program" type="python">
ready = True
if ready:
    print("Hello World!")
          </sourcecode>
        </figure>
      </section>
    </section>
  </middle>
  <back>
    <references anchor="_normative_references">
      <name>Normative References</name>
      <xi:include
href="http://xml2rfc.ietf.org/public/rfc/bibxml/reference.RFC.2119.xml"
parse="text" />
    </references>
    <references anchor="_informative_references">
      <name>Informative References</name>
      <xi:include
href="http://xml2rfc.ietf.org/public/rfc/bibxml/reference.RFC.2635.xml"
parse="text" />
    </references>
  </back>
</rfc>
```

23. References

23.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC7991] Hoffman, P., "The "xml2rfc" Version 3 Vocabulary", RFC 7991, DOI 10.17487/RFC7991, December 2016, <https://www.rfc-editor.org/info/rfc7991>.

23.2. Informative References

[AsciiDoc] Rackham, S., "AsciiDoc: Text based document generation", November 2013, <http://www.methods.co.nz/asciidoc/>.

[Asciidoctor] Allen, D., Waldron, R., and S. White, "Asciidoctor: A fast text processor & publishing toolchain for converting AsciiDoc to HTML5, DocBook & more.", November 2017, <http://asciidoctor.org>.

[asciidoctor-bibliography] Ribose Inc., "Citations and Bibliography the 'asciidoctor-way'", November 2017, <https://github.com/riboseinc/asciidoctor-bibliography/>.

[Asciidoctor-Manual] Allen, D., Waldron, R., and S. White, "Asciidoctor: A fast text processor & publishing toolchain for converting AsciiDoc to HTML5, DocBook & more.", November 2017, <http://asciidoctor.org/docs/user-manual/>.

[asciidoctor-rfc] Ribose Inc., "asciidoctor-rfc lets you write Internet-Drafts and RFCs in AsciiDoc, the "asciidoctor-way".", November 2017, <https://github.com/riboseinc/asciidoctor-rfc/>.

[AsciiMathML] "AsciiMath is an easy-to-write markup language for mathematics.", November 2017, <http://asciimath.org>.

[draftr] Barnes, R., "draftr: an HTML front-end to pandoc2rfc", Nov 2017, <https://ipv.sx/draftr/>.

[IETF-BibXML] "IETF BibXML Library", November 2017, <http://xml.resource.org/public/rfc/bibxml/>.

[kramdown-rfc2629] Bormann, C., "kramdown-rfc2629: An RFC2629 (XML2RFC) backend for Thomas Leitner's kramdown markdown parser", Nov 2017, <https://github.com/cabo/kramdown-rfc2629>.

[lyx2rfc] Williams, N., "LyX to I-D/RFC export by way of Lyx export to XHTML and XSLT conversion to xml2rfc schema", 2014, <https://github.com/nicowilliams/lyx2rfc>.

[MathJax] "MathJax: A JavaScript display engine for mathematics that works in all browsers.", November 2017, <https://www.mathjax.org>.

[mmark] Gieben, R., "Using mmark to create I-Ds and RFCs", June 2015, <https://github.com/miekg/mmark>.

[NroffEdit] Santesson, S., "WYSIWYG Internet-Draft Nroff Editor", May 2011, <http://aaa-sec.com/nroffedit/>.

[pandoc2rfc] Gieben, R., "pandoc2rfc: Use pandoc to create XML suitable for xml2rfc", 2012, <https://github.com/miekg/pandoc2rfc>.

[RFC5385] Touch, J., "Version 2.0 Microsoft Word Template for Creating Internet Drafts and RFCs", RFC 5385, DOI 10.17487/RFC5385, February 2010, <https://www.rfc-editor.org/info/rfc5385>.

[RFC7328] Gieben, R., "Writing I-Ds and RFCs Using Pandoc and a Bit of XML", RFC 7328, DOI 10.17487/RFC7328, August 2014, <https://www.rfc-editor.org/info/rfc7328>.

[RFC7749] Reschke, J., "The "xml2rfc" Version 2 Vocabulary", RFC 7749, DOI 10.17487/RFC7749, February 2016, <https://www.rfc-editor.org/info/rfc7749>.

[RFC7763] Leonard, S., "The text/markdown Media Type", RFC 7763, DOI 10.17487/RFC7763, March 2016, <https://www.rfc-editor.org/info/rfc7763>.

[RFC7764] Leonard, S., "Guidance on Markdown: Design Philosophies, Stability Strategies, and Select Registrations", RFC 7764, DOI 10.17487/RFC7764, March 2016, <https://www.rfc-editor.org/info/rfc7764>.

[RFC7990] Flanagan, H., "RFC Format Framework", RFC 7990, DOI 10.17487/RFC7990, December 2016, <https://www.rfc-editor.org/info/rfc7990>.

[TeX-LaTeX] "LaTeX is document preparation software that runs on top of Donald E. Knuth's TeX typesetting system.", November 2017, <https://www.latex-project.org>.

Appendix A. Acknowledgements

The authors would like to thank the following persons for their valuable advice and input.

Authors' Addresses

Ronald Henry Tse
Ribose
Suite 1111, 1 Pedder Street
Central, Hong Kong
Hong Kong

Email: ronald.tse@ribose.com
URI: https://www.ribose.com

Nick Nicholas
Ribose
Australia

Email: nick.nicholas@ribose.com
URI: https://www.ribose.com

Jeffrey Lau
Ribose
Suite 1111, 1 Pedder Street
Central, Hong Kong
Hong Kong

Email: jeffrey.lau@ribose.com
URI: https://www.ribose.com

Paolo Brasolin
Ribose
Italy

Email: paolo.brasolin@ribose.com
URI: https://www.ribose.com