Network Working Group
Internet-Draft
Intended status: Informational
Expires: July 4, 2017

E. Wilde
CA Technologies

December 31, 2016

A Media Type Structured Syntax Suffix for JSON Text Sequences
draft-wilde-json-seq-suffix-03

Abstract

Structured Syntax Suffixes for media types allow other media types to build on them and make it explicit that they are built on an existing media type as their foundation. This specification defines and registers "+json-seq" as a structured syntax suffix for JSON Text Sequences.

Note to Readers

[[The RFC Editor is requested to remove this section at publication.]]

This draft should be discussed on the ietf mailing list (<https://www.ietf.org/mailman/listinfo/ietf>).

Online access to all versions and files is available on GitHub (<https://github.com/dret/I-D/tree/master/json-seq-suffix>).

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on July 4, 2017.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

Media Type Structured Syntax Suffixes [RFC6838] were introduced as a way for a media type to signal that it is based on another media type as its foundation. Some structured syntax suffixes were registered initially [RFC6839], including "+json" for the widely popular JSON Format [RFC7159].

JSON Text Sequences [RFC7464] is a recent specification in the JSON space that defines how a sequence of multiple JSON texts can be represented in one representation. This document defines and registers the "+json-seq" structured syntax suffix in the Structured Syntax Suffix Registry.

2. Terminology

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].

3. The "+json-seq" Structured Syntax Suffix

The use case for the "+json-seq" structured syntax suffix is the same as for "+json": It SHOULD be used by media types when parsing the JSON Text Sequence of a media type leads to a meaningful result, by simply using the generic JSON Text Sequence processing.

Applications encountering such a media type can then either simply use generic processing if all they need is a generic view of the JSON Text Sequence, or they can use generic JSON Text Sequence tools for initial parsing, and then can implement their own specific processing on top of that generic parsing tool.

4. IANA Considerations

Structured Syntax Suffixes are registered within the "Structured Syntax Suffix Registry" maintained at <https://www.iana.org/assignments/media-type-structured-suffix>.

IANA is requested to register the "+json-seq" structured syntax suffix in accordance with [RFC6838].

Name: JSON Text Sequence

+suffix: +json-seq

References: [RFC7464], RFC [[RFC Editor, please insert assigned RFC number.]]

Encoding considerations: See [RFC7464] Section 2.2

Fragment identifier considerations: The syntax and semantics of fragment identifiers specified for +json-seq SHOULD be as specified for "application/json-seq". (At publication of this document, there is no fragment identification syntax defined for "application/json-seq".)

The syntax and semantics for fragment identifiers for a specific "xxx/yyy+json-seq" SHOULD be processed as follows:

For cases defined in +json-seq, where the fragment identifier resolves per the +json-seq rules, then process as specified in +json-seq.

For cases defined in +json-seq, where the fragment identifier does not resolve per the +json-seq rules, then process as specified in "xxx/yyy+json-seq".

For cases not defined in +json-seq, then process as specified in "xxx/yyy+json-seq".

Interoperability considerations: n/a

Security considerations: See [RFC7464] Section 3

Contact: Applications and Real-Time Area Discussion (art@ietf.org), or any IESG designated successor.

Author/Change controller: The Applications and Real-Time Area Working Group. IESG has change control over this registration.

5. Security Considerations

All the security considerations of JSON Text Sequences [RFC7464] apply. They are as follows:

All the security considerations of JSON [RFC7159] apply. This format provides no cryptographic integrity protection of any kind.

As usual, parsers must operate on input that is assumed to be untrusted. This means that parsers must fail gracefully in the face of malicious inputs.

Note that incremental JSON text parsers can produce partial results and later indicate failure to parse the remainder of a text. A sequence parser that uses an incremental JSON text parser might treat a sequence like '<RS>"foo"<LF>456<LF><RS>' as a sequence of one element ("foo"), while a sequence parser that uses a non-incremental JSON text parser might treat the same sequence as being empty. This effect, and texts that fail to parse and are ignored, can be used to smuggle data past sequence parsers that don't warn about JSON text failures.

Repeated parsing and re-encoding of a JSON text sequence can result in the addition (or stripping) of trailing LF bytes from (to) individual sequence element JSON texts. This can break signature validation. JSON has no canonical form for JSON texts, therefore neither does the JSON text sequence format.

6. References

6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <http://www.rfc-editor.org/info/rfc6838>.

[RFC7464] Williams, N., "JavaScript Object Notation (JSON) Text Sequences", RFC 7464, DOI 10.17487/RFC7464, February 2015, <http://www.rfc-editor.org/info/rfc7464>.

6.2. Informative References

[RFC6839] Hansen, T. and A. Melnikov, "Additional Media Type Structured Syntax Suffixes", RFC 6839, DOI 10.17487/RFC6839, January 2013, <http://www.rfc-editor.org/info/rfc6839>.

[RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <http://www.rfc-editor.org/info/rfc7159>.

Appendix A. Acknowledgements

Thanks for comments and suggestions provided by Ben Campbell, Allan Doyle, Warren Kumari, Sean Leonard, Alexey Melnikov, Brian Raymor, and Peter Yee.

Author's Address

Erik Wilde
CA Technologies

Email: erik.wilde@dret.net
URI: http://dret.net/netdret/