Applications Area Working Group
Internet-Draft
Intended status: Informational
Expires: September 10, 2012

P. Bryan, Ed.
ForgeRock
K. Zyp
SitePen (USA)

March 9, 2012

JSON Pointer
draft-ietf-appsawg-json-pointer-01

Abstract

JSON Pointer defines a string syntax for identifying a specific value within a JSON document.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

This specification defines JSON Pointer, a string syntax for identifying a specific value within a JavaScript Object Notation (JSON) [RFC4627] text document. This syntax is intended to be easily expressed in JSON string values and Uniform Resource Identifier (URI) [RFC3986] fragment identifiers.

2. Conventions

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].

This specification expresses normative syntax rules using Augmented Backus-Naur Form [RFC5234] (ABNF) notation.

3. Syntax

A JSON Pointer is a [Unicode] string containing a sequence of zero or more reference tokens, each prefixed by a '/' (%x2F) character.

If a reference token contains '/' (%x2F) or '^' (%x5E) characters, such characters MUST each be prefixed (escaped) with a '^' (%x5E) character.

ABNF syntax:

```
json-pointer = *( "/" reference-token )
reference-token = *( unescaped / escaped )
unescaped = %x00-2E / %x30-5B / %x5D-10FFFF
escaped = "^" ( "/" / "^" )
```

It is an error condition if a JSON Pointer value does not conform to this syntax.

4. Evaluation

Evaluation of a JSON Pointer begins with a reference to the root value of a JSON text document and completes with a reference to some value within the document. Each reference token in the JSON Pointer is sequentially evaluated.

Evaluation of each reference token begins by unescaping any escaped character sequence; this is performed by removing the '^' (escape)

prefix. The reference token then modifies which value is referenced according to the following scheme:

> If the currently referenced value is a JSON object, the new referenced value is the object member with the name identified by the reference token. The member name is equal to the token if it has the same number of Unicode characters as token and their codepoints are positionwise equal. If a referenced member name is not unique in an object, the member that is referenced is undefined.
>
> If the currently referenced value is a JSON array, the reference token MUST contain characters that represent an unsigned base-10 integer value, and the new referenced value is the array element with the zero-based index identified by the token.

If a reference token is being evaluated against a JSON document, the implementation MAY evaluate each token against a concrete value, and terminate evaluation with an error condition if a evaluation fails to resolve a concrete value.

5. JSON String Representation

A JSON Pointer MAY be represented in a JSON string value. Per [RFC4627], section 2.5, all instances of quotation mark '"' (%x22), reverse solidus '\' (%x5C) and control (%x00-1F) characters MUST be escaped.

6. URI Fragment Identifier Representation

A JSON Pointer MAY be represented in a URI fragment identifier. The JSON pointer MUST be UTF-8 [RFC3629] encoded as octets; octets not in the URI "unreserved" set SHOULD be percent-encoded, per [RFC3986], section 2.5.

7. Error Handling

In the event of an error condition, evaluation of the JSON Pointer fails to complete.

8. IANA Considerations

This document has no IANA actions.

9. Security Considerations

A given JSON Pointer is not guaranteed to reference an actual JSON value. Implementations should be aware of this and take appropriate precautions.

10. Acknowledgements

The following individuals contributed ideas, feedback and wording, which contributed to the content of this specification:

Mike Acar, Carsten Bormann, Tim Bray, Jacob Davies, Martin J. Duerst, Bjoern Hoehrmann, James H. Manger, Mark Nottingham, Drew Perttula, Julian Reschke.

11. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4627] Crockford, D., "The application/json Media Type for JavaScript Object Notation (JSON)", RFC 4627, July 2006.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[Unicode] The Unicode Consortium, "The Unicode Standard, Version 6.0", October 2011, <http://www.unicode.org/versions/Unicode6.0.0/>.

Appendix A. Examples

The following examples illustrate the use of JSON Pointers in URI fragments for a JSON text document located at http://example.com/example.json, with the following value:

```
{
    "foo": {
        "bar": [ "element0", "element1" ],
        "inner object": {
            "baz": "qux"
    }
}
```

http://example.com/example.json#
Resolves to the object value at the root of the JSON text document.

http://example.com/example.json#/foo
Resolves to the object value of the "foo" member in the root object.

http://example.com/example.json#/foo/inner%20object
Resolves to the object value of the "inner object" member in the "foo" object value in the root object.

http://example.com/example.json#/foo/inner%20object/baz
Resolves to the string value "qux", which is the value of the "baz" member in the "inner object" member in the "foo" member in the root object.

http://example.com/example.json#/foo/bar/0
Resolves to the string value "element0", which is the first value in the "bar" array in the "foo" member in the root object.

Authors' Addresses

Paul C. Bryan (editor)
ForgeRock

Phone: +1 604 783 1481
Email: pbryan@anode.ca

Kris Zyp
SitePen (USA)

Phone: +1 650 968 8787
Email: kris@sitepen.com