Workgroup: Internet Engineering Task Force
Internet-Draft:
draft-hha-relative-json-pointer-00
Published: 19 June 2023
Intended Status: Informational
Expires: 21 December 2023

G. Luff
H. Andrews, Ed.
B. Hutton, Ed.

Relative JSON Pointers

Abstract

JSON Pointer is a syntax for specifying locations in a JSON document, starting from the document root. This document defines an extension to the JSON Pointer syntax, allowing relative locations from within the document.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

JSON Pointer (RFC 6901 [RFC6901]) is a syntax for specifying locations in a JSON document, starting from the document root. This document defines a related syntax allowing identification of relative locations from within the document.

2. Conventions and 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 RFC 2119 [RFC2119].

3. Syntax

A Relative JSON Pointer is a Unicode string in UTF-8 encoding (see RFC 8259, Section 8 [RFC8259]), comprising a non-negative integer, an optional index adjustment consisting of '+' (%x2B) or '-' (%x2D) followed by a positive integer, followed by either a '#' (%x23) character or a JSON Pointer (RFC 6901 [RFC6901]).

The separation between the integer prefix (with optional adjustment) and the JSON Pointer will always be unambiguous, because a JSON Pointer must be either zero- length or start with a '/' (%x2F). Similarly, a JSON Pointer will never be ambiguous with the '#'.

The ABNF syntax of a Relative JSON Pointer is:

```
relative-json-pointer = origin-specification ( "#" / json-pointer )
                      ; json-pointer from RFC 6901

origin-specification  = non-negative-integer [ index-manipulation ]
index-manipulation    = ( "+" / "-" ) positive-integer
non-negative-integer  = "0" / positive-integer
positive-integer      = %x31-39 *DIGIT
                      ; digits without a leading zero
```

4. Evaluation

Evaluation of a Relative JSON Pointer begins with a reference to a value within a JSON document, and completes with either a value within that document, a string corresponding to an object member, or integer value representing an array index.

Evaluation begins by processing the non-negative-integer prefix. This can be found by taking the longest continuous sequence of decimal digits available, starting from the beginning of the string, taking the decimal numerical value. If the value is zero, the following steps are skipped. If this value is more than zero, then the following steps are repeated that number of times:

> If the current referenced value is the root of the document, then evaluation fails (see below).
>
> If the referenced value is an item within an array, then the new referenced value is that array.
>
> If the referenced value is an object member within an object, then the new referenced value is that object.

If the next character is a plus ("+") or minus ("-"), followed by another continuous sequence of decimal digits, the following steps are taken using the decimal numeric value of that plus or minus sign and decimal sequence:

> If the current referenced value is not an item of an array, then evaluation fails (see below).
>
> If the referenced value is an item of an array, then the new referenced value is the item of the array indexed by adding the decimal value (which may be negative), to the index of the current referenced value.

If the remainder of the Relative JSON Pointer is a JSON Pointer, then evaluation proceeds as per RFC 6901, Section 5 [RFC6901] with the modification that the initial reference being used is the reference currently being held (which may not be root of the document).

Otherwise (when the remainder of the Relative JSON Pointer is the character '#'), the final result is determined as follows:

> If the current referenced value is the root of the document, then evaluation fails (see below).
>
> If the referenced value is an item within an array, then the final evaluation result is the value's index position within the array.
>
> If the referenced value is an object member within an object, then the new referenced value is the corresponding member name.

5. JSON String Representation

The concerns surrounding JSON String representation of a Relative JSON Pointer are identical to those laid out in RFC 6901, Section 5 [RFC6901].

5.1. Examples

For example, given the JSON document:

```
{
   "foo": ["bar", "baz", "biz"],
   "highly": {
      "nested": {
         "objects": true
      }
   }
}
```

Starting from the value "baz" (inside "foo"), the following JSON strings evaluate to the accompanying values:

```
"0"                         "baz"
"1/0"                       "bar"
"0-1"                       "bar"
"2/highly/nested/objects"   true
"0#"                        1
"0+1#"                      2
"1#"                        "foo"
```

Starting from the value {"objects":true} (corresponding to the member key "nested"), the following JSON strings evaluate to the accompanying values:

```
"0/objects"              true
"1/nested/objects"       true
"2/foo/0"                "bar"
"0#"                     "nested"
"1#"                     "highly"
```

6. Non-use in URI Fragment Identifiers

Unlike a JSON Pointer, a Relative JSON Pointer can not be used in a URI fragment identifier. Such fragments specify exact positions within a document, and therefore Relative JSON Pointers are not suitable.

7. Error Handling

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

Evaluation may fail due to invalid syntax, or referencing a non-existent value. This specification does not define how errors are handled. An application of JSON Relative Pointer SHOULD specify the impact and handling of each type of error.

8. Relationship to JSON Pointer

Relative JSON Pointers are intended as a companion to JSON Pointers. Applications MUST specify the use of each syntax separately. Defining either JSON Pointer or Relative JSON Pointer as an acceptable syntax does not imply that the other syntax is also acceptable.

9. Acknowledgements

The language and structure of this specification are based heavily on [RFC6901], sometimes quoting it outright.

This draft remains primarily as written and published by Geraint Luff, with only minor subsequent alterations under new editorship.

10. Security Considerations

Evaluation of a given Relative JSON Pointer is not guaranteed to reference an actual JSON value. Applications using Relative JSON Pointer should anticipate this situation by defining how a pointer that does not resolve ought to be handled.

As part of processing, a composite data structure may be assembled from multiple JSON documents (in part or in full). In such cases, applications SHOULD ensure that a Relative JSON Pointer does not evaluate to a value outside the document for which is was written.

Note that JSON pointers can contain the NUL (Unicode U+0000) character. Care is needed not to misinterpret this character in programming languages that use NUL to mark the end of a string.

11. IANA Considerations

This document has no IANA actions.

12. References

12.1. 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>.

[RFC6901] Bryan, P., Ed., Zyp, K., and M. Nottingham, Ed., "JavaScript Object Notation (JSON) Pointer", RFC 6901, DOI 10.17487/RFC6901, April 2013, <https://www.rfc-editor.org/info/rfc6901>.

12.2. Informative References

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

Appendix A. ChangeLog

This section to be removed before leaving Internet-Draft status.

draft-hha-relative-json-pointer-00 Fix ABNF omission for using # with index manipulation

- Clarify handling of leading "0"

draft-bhutton-relative-json-pointer-00 Add array forward and backward index manipulation

draft-handrews-relative-json-pointer-02 Update to the latest JSON RFC

draft-handrews-relative-json-pointer-01 The initial number is "non-negative", not "positive"

draft-handrews-relative-json-pointer-00 Revived draft with identical wording and structure.

- Clarified how to use alongside JSON Pointer.

draft-luff-relative-json-pointer-00 Initial draft.

Authors' Addresses

Geraint Luff
Cambridge
United Kingdom

Email: luffgd@gmail.com

Henry H. Andrews (editor)

Email: andrews_henry@yahoo.com

Ben Hutton (editor)

Email: ben@jsonschema.dev
URI: https://jsonschema.dev