Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: June 6, 2012

P. Bryan, Ed.
ForgeRock US, Inc.

December 4, 2011

JSON Patch
draft-pbryan-json-patch-03

Abstract

JSON Patch defines the media type "application/json-patch", a JSON document structure for expressing a sequence of partial modifications to a JSON document.

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 June 6, 2012.

Table of Contents

1. Introduction

JavaScript Object Notation (JSON) [RFC4627] is a common format for the exchange and storage of structured data. HTTP PATCH [RFC5789] extends HTTP [RFC2616] with a method to perform partial modifications to resources.

The JSON Patch media type "application/json-patch" is a JSON document structure for expressing a sequence of partial modifications to a JSON document, suitable for use with the HTTP PATCH method.

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 RFC 2119 [RFC2119].

3. Document Structure

A JSON Patch document contains a JSON array of objects. Each object contains a single operation to apply to the target JSON document.

A sample JSON Patch document:

```
[
    { "test": "/a/b/c", value: "foo" },
    { "remove": "/a/b/c" },
    { "add": "/a/b/c", "value": [ "foo", "bar" ] },
    { "replace": "/a/b/c", "value": 42 },
    { "move": "/a/b/c", to: "/a/b/d" }
]
```

Evaluation of a JSON Patch document begins with a target JSON document to modify. Operations are applied sequentially in the order they appear in the array. Each operation in the sequence is applied to the target document. The resulting modified document becomes the target for the next operation. The process repeats until all operations are successfully applied.

4. Operations

The operation to perform is expressed in the name of a member in the operation object; it's value is a string containing a [JSON Pointer], which references the value for which to apply the operation. It is an error condition if an operation object contains more than one

operation member.

4.1. add

The "add" operation adds a new value into the target document. The value to be added is specified in the operation object's "value" member.

If adding to an object, it is an error condition if the member to be added in the object already exists.

If adding to an array, all elements at or above the specified index are shifted one position to the right. It is an error condition if the specified index is greater than the number of elements in the existing array.

4.2. remove

The "remove" operation removes a value from the target document.

If removing an element from an array, all elements above the specified index are shifted one position to the left.

It is an error condition if the value to be removed does not exist.

4.3. replace

The "replace" operation replaces an existing value in the target document with a new value. The value to replace the existing value with is specified in the operation object's "value" member.

This operation is semantically equivalent to expressing a "remove" operation for a value, followed immediately by an "add" operation at the same location of the removed value.

It is an error condition if the value to be replaced does not exist.

4.4. move

The "move" operation moves a value from one location to another within the target document. The location to move the value to is specified in the operation object's "to" member, a string containing a JSON Pointer.

This operation is semantically equivalent to expressing a "remove" operation, followed immediately by an "add" operation with the removed value.

It is an error condition if the value to be moved does not exist, or if the location to move the value to already has a value.

4.5. test

The "test" operation tests that a value in the target document is equal to the specified value. The value to test for is specified in the operation object's "value" member.

It is an error condition if the value in the target document is not equal to the specified value.

5. Error Handling

In the event of an error condition, evaluation of the JSON Patch document SHOULD terminate and application of the entire patch document MUST NOT be deemed successful.

6. IANA Considerations

The Internet media type for a JSON Patch document is application/json-patch.

Type name: application

Subtype name: json-patch

Required parameters: none

Optional parameters: none

Encoding considerations:
Per JSON [RFC4627]: 8bit if UTF-8; binary if UTF-16 or UTF-32.

Security considerations:
See Security Considerations in section 8.

Interoperability considerations: N/A

Published specification:
draft-pbryan-json-patch-03

Applications that use this media type:
Applications that manipulate JSON documents.

Additional information:

Magic number(s): N/A

File extension(s): .json-patch

Macintosh file type code(s): TEXT

Person & email address to contact for further information:
Paul C. Bryan <paul.bryan@forgerock.com>

Intended usage: COMMON

Restrictions on usage: none

Author: Paul C. Bryan <paul.bryan@forgerock.com>

Change controller: Paul C. Bryan <paul.bryan@forgerock.com>

7. Security Considerations

This specification has the same security considerations as JSON [RFC4627] and JSON Pointer [JSON Pointer].

8. Acknowledgements

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

Mike Amundsen, Paul Davis, Dean Landolt, Randall Leeds, Mark Nottingham, Julian Reschke, Eli Stevens.

The structure of a JSON Patch document was initially informed by the XML Patch document [RFC5261] specification.

9. References

9.1. Normative References

[JSON Pointer] Bryan, P. and K. Zyp, "JSON Pointer", October 2011, <http://tools.ietf.org/html/draft-pbryan-zyp-json-pointer-02>.

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

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

9.2. Informative References

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC5261] Urpalainen, J., "An Extensible Markup Language (XML) Patch Operations Framework Utilizing XML Path Language (XPath) Selectors", RFC 5261, September 2008.

[RFC5789] Dusseault, L. and J. Snell, "PATCH Method for HTTP", RFC 5789, March 2010.

Appendix A. Examples

A.1. Adding an Object Member

An example target JSON document:

```
{
    "foo": "bar"
}
```

A JSON Patch document:

```
[
    { "add": "/baz", "value": "qux" }
]
```

The resulting JSON document:

```
{
    "baz": "qux",
    "foo": "bar"
}
```

A.2. Adding an Array Element

An example target JSON document:

```
{
    "foo": [ "bar", "baz" ]
}
```

A JSON Patch document:

```
[
    { "add": "/foo/1", "value": "qux" }
]
```

The resulting JSON document:

```
{
    "foo": [ "bar", "qux", "baz" ]
}
```

A.3. Removing an Object Member

An example target JSON document:

```
{
    "baz": "qux",
    "foo": "bar"
}
```

A JSON Patch document:

```
[
    { "remove": "/baz" }
]
```

The resulting JSON document:

```
{
    "foo": "bar"
}
```

A.4. Removing an Array Element

An example target JSON document:

```
{
    "foo": [ "bar", "qux", "baz" ]
}
```

A JSON Patch document:

```
[
    { "remove": "/foo/1" }
]
```

The resulting JSON document:

```
{
    "foo": ["bar", "baz"]
}
```

A.5. Replacing a Value

An example target JSON document:

```
{
    "baz": "qux",
    "foo": "bar"
}
```

A JSON Patch document:

```
[
      { "replace": "/baz", "value": "boo" }
]
```

The resulting JSON document:

```
{
    "baz": "boo",
    "foo": "bar"
}
```

A.6. Moving a Value

An example target JSON document:

```
{
    "foo": {
        "bar": "baz",
        "waldo": "fred"
    }
    "qux": {
        "corge": "grault"
    }
}
```

A JSON Patch document:

```
[
    { "move": "/foo/waldo", to: "/qux/thud" }
]
```

The resulting JSON document:

```
{
    "foo": {
        "bar": "baz"
    }
    "qux": {
        "corge": "grault",
        "thud": "fred"
    }
}
```

A.7. Moving an Array Element

An example target JSON document:

```
{
    "foo": [ "all", "grass", "cows", "eat" ]
}
```

A JSON Patch document:

```
[
    { "move": "/foo/1", "to": "/foo/3" }
]
```

The resulting JSON document:

```
{
    "foo": [ "all", "cows", "eat", "grass" ]
}
```

A.8. Testing a Value: Success

An example target JSON document:

```
{
    "baz": "qux",
    "foo": [ "a", 2, "c" ]
}
```

A JSON Patch document, which will result in successful evaluation:

```
[
    { "test": "/baz", "value": "qux" },
    { "test": "/foo/1", "value": 2 }
]
```

A.9. Testing a Value: Error

An example target JSON document:

```
{
    "baz": "qux",
}
```

A JSON Patch document, which will result in an error condition:

```
[
    { "test": "/baz", "value": "bar" }
]
```

Author's Address

Paul C. Bryan (editor)
ForgeRock US, Inc.
201 NE Park Plaza Drive Suite 196
Vancouver, WA 98684
USA

Phone: +1 604 783 1481
Email: paul.bryan@forgerock.com