Network Working Group
Internet-Draft
Intended status: Informational
Expires: December 10, 2012

J. Snell
June 8, 2012

The application/merge-patch Media Type
draft-snell-merge-patch-02

Abstract

This specification defines the application/merge-patch media type and it's intended use with the HTTP PATCH method defined by RFC 5789; as well as the "application/json+merge-patch" variation that defines the "Merge Patch" semantics for JSON documents.

Status of this Memo

This Internet-Draft is submitted to IETF 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 December 10, 2012.

Table of Contents

1. Introduction

The HTTP PATCH method [RFC5789] provides a mechanism for requesting partial modifications of resources. The payload entity contained by a PATCH request provides a description of the changes that are to be made to the target resource. The general term used to describe such payloads is a "Patch Document".

A partial modification request using PATCH can generally take one of two forms. The Patch Document can either

- Provide an explicit description of the changes being requested -- as is done, for instance, with the JSON Patch format described in [I-D.ietf-appsawg-json-patch] -- or,
- Provide a modified version of the original resource and allow the Server to determine to specific set of changes being requested.

Either approach is equally valid. However, it is important to note that when using PATCH with the second approach -- generally called a "Merge Patch" -- it is often difficult for a server to determine the clients exact intent when generic media types that do not have clearly defined PATCH semantics defined are used.

To best illustrate the problem -- albeit with an example that is somewhat extreme -- consider an example where we have the following JSON Patch Document currently existing on a server that we wish to modify:

```
[
 {"add":"title","value":"Goodbye!"},
 {"remove":"link"}
]
```

If we send the following request to the server intending to perform a Merge Patch style modification:

```
PATCH /patches/1 HTTP/1.1
Host: example.org
Content-Type: application/json-patch

[{"add":"title","value":"Hello world"}]
```

The server has no choice but to interpret this request as a normal JSON Patch operation, resulting in an unintended modification of the target resource.

What is needed in this case is a mechanism that will allow the user agent sending the PATCH request to explicitly signal that it is requesting a Merge Patch style modification of the resource.

Using the "application/merge-patch" Media Type defined herein, the user agents original intent can be clearly and unambiguously communicated to the server within the request:

```
PATCH /patches/1 HTTP/1.1
Host: example.org
Content-Type: application/merge-patch;
  type="application/json+patch"; charset="UTF-8"

[{"add":"title","value":"Hello world"}]
```

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 [RFC2119].

2. The "application/merge-patch" Media Type

The "application/merge-patch" Media Type is used to identify document resources that describe, by example, a set of changes that are to be made to a target resource. When used within an HTTP PATCH request, it is the responsibility of the server receiving and processing the request to inspect the payload entity and determine the specific set of operations that are to be performed to modify the target resource. The actual set of modifications to be made will be specific to the media type of the target resource.

For example, given the following example JSON document:

```
{
  "title": "Goodbye!",
  "author" : {
    "givenName" : "James",
    "familyName" : "Snell"
  },
  "tags":["example","sample"]
}
```

If my intent is to change only the value of the "title" property from it's current value "Goodbye!" to the new value "Hello!", I would send the following request:

```
PATCH /my/resource HTTP/1.1
Host: example.org
Content-Type: application/merge-patch;
  type="application/json"; charset="UTF-8""

{
  "title": "Hello!"
}
```

Upon receiving the request, the server is responsible for inspecting the payload and will determine, based on it's own understanding of the target resource media type and the underlying data model the target resource represents, what specific operations will be applied to modify the resource.

The "application/merge-patch" media type intentionally makes the receiving party responsible for determining how the target resource is to be modified.

2.1. The "type" Parameter

By itself, the "application/merge-patch" media type does not define a specific serialization format and MAY contain data of any type.

Unless using a type-specific variation (Section 2.3) of the "application/merge-patch" media type, such as "application/json+merge-patch" (Section 3), the "application/merge-patch" MUST specify a type parameter whose value is the actual MIME Media Type of the payload.

For example, if using the "application/merge-patch" media type to request modifications in an XML document, the "type" parameter would specify the appropriate XML media type:

```
PATCH /my/resource HTTP/1.1
Host: example.org
Content-Type: application/merge-patch;
  type="application/xml"; charset="UTF-8"

<abc>
  <xyz>foo</xyz>
</abc>
```

2.2. The "charset" Parameter

When the "application/merge-patch" resource contains character-based data (e.g. plain text, XML, JSON, and so forth), the "charset" parameter SHOULD be used to identify the character set encoding

utilized.

2.3. Type-Specific Variations

This document recommends the use of a naming convention (a suffix of "+merge-patch") for identifying Media Type specific variations of "application/merge-patch".

Media Type specific variations of "application/merge-patch" define the specific details of how a server SHOULD derive the set of actual change operations being requested in a PATCH operation relative to one or more specific media types. The variation also defines the specific payload type required for the request.

When a Media Type specific variation is used, such as "application/json+merge-patch", the "type" parameter MAY be specified to provide additional specific type information.

In the following example, the request clearly indicates that the "application/json+merge-patch" Type Specific Variation is utilized with the "type" attribute providing additional contextual information about what specific variation of the JSON Document format is contained in the request.

```
PATCH /my/resource HTTP/1.1
Host: example.org
Content-Type: application/json+merge-patch;
  type="application/stream+json"; charset="UTF-8"

{
  "verb": "POST"
}
```

3. The "application/json+merge-patch" Type Specific Variation

The "application/json+merge-patch" Media Type is a Type Specific Variation of the "application/merge-patch" Media Type that uses a JSON data structure to describe the changes to be made to a target resource.

For example, given the following example JSON document:

```
{
  "title": "Goodbye!",
  "author" : {
    "givenName" : "James",
    "familyName" : "Snell"
  },
  "tags":["example","sample"]
}
```

If the intent is to change the value of the "title" property to from "Goodbye!" to the value "Hello!", add a new "phoneNumber" property, remove the "familyName" property from the "author" object, and remove the word sample from the "tags" Array, the user-agent would send the following request:

```
PATCH /my/resource HTTP/1.1
Host: example.org
Content-Type: application/json+merge-patch; charset="UTF-8"

{
  "title": "Hello!",
  "phoneNumber": "+01-123-456-7890",
  "author": {
    "familyName": null
  }
  "tags": ["example"]
}
```

Note that while the payload of "application/json+merge-patch" resources MUST be any valid subtype of the "application/json" media type, the target resource to which a PATCH request using "application/json+merge-patch" is sent can be of any arbitrary media type. It is the server's responsibility to determine how to apply the specific semantics of the Merge Patch operation to the target resource.

A server receiving this patch request MUST determine the specific set of change operations to be performed by analyzing the JSON data contained in the payload of the request relative to the target resource and applying the following rules:

1. If the root of the JSON data provided in the payload is a JSON Array, the target resource is to be replaced, in whole, by the provided JSON data.
2. If the root of the JSON data provided in the payload is a JSON Object, for each distinct property specified in that object:

* If the property is currently not specified for the target resource, the property and the given value is to be added to the target.
* If the value is explicitly set to null and that property is currently specified for the target resource, the existing value is removed.
* If the value is either a non-null JSON primitive or an Array and that property is currently specified for the target resource, the existing value is to be replaced with that provided.
* If the value is a JSON object and that property is currently specified for the target resource and the existing value is a JSON primitive or Array, the existing value is to be replaced in whole by the object provided.
* If the value is a JSON object and that property is currently specified for the target resource and the existing value is also a JSON object, then recursively apply Rule #2 to each object.

Applying these rules to the previous example, the set of specific change operations the server would derive from the request would be:

- Change the existing value of the "title" property from "Goodbye!" to "Hello!",
- Add the "phoneNumber" property with a value of "+01-123-456-7890",
- Remove the "familyName" property from the current object value associated with the "author" property, and
- Change the existing value of the "tags" property from ["example","sample"] to ["example"].

Note that once the set of intended modifications is derived from the request, the server is free to determine the appropriateness of the modification based on it's own understanding of the target resource. For instance, in the previous example, it is possible that the "familyName" property could be required within the target resource and cannot be removed. Note that in such cases, per [RFC5789], Section 2, the server is REQUIRED to reject the PATCH request using an HTTP error response code appropriate to the error condition.

If the request attempts to remove a property from the target resource that does not currently exist, the server SHOULD NOT consider the request to be in error. The requested removal operation SHOULD simply be ignored by the server as the final modified state of the target resource will still accurately reflect the user-agent's original intention.

4. IANA Considerations

This specification registers the following additional MIME Media Types:

4.1. application/merge-patch

Type name: application
Subtype name: merge-patch
Required parameters: "type" : A MIME Media Type
Optional parameters: "charset" : Specifies the character set encoding type when the "application/merge-patch" resource contains character-based content.
Encoding considerations: The specific encoding considerations will vary dependent on the specific MIME Media Type specified by the "type" parameter.
Security considerations: As defined in this specification.
Interoperability considerations: There are no known interoperability issues.
Published specification: This specification.
Applications that use this media type: This media type is intended primarily for use with the HTTP PATCH method.
Additional information:
   Magic number(s): N/A
   File extension(s): N/A
   Macintosh file type code(s): "BINARY"
Person & email address to contact for further information: James M Snell <jasnell@gmail.com>
Intended usage: COMMON
Restrictions on usage: None.
Author: James M Snell <jasnell@gmail.com>
Change controller: IETF

4.2. application/json+merge-patch

Type name: application
Subtype name: json+merge-patch
Required parameters: None
Optional parameters: "charset" : Specifies the character set encoding. If not specified, a default of "UTF-8" is assumed.
Encoding considerations: Resources that use the "application/json+merge-patch" media type are required to conform to the "application/json" Media Type and are therefore subject to the same encoding considerations specified in Section 6 [RFC4627].
Security considerations: As defined in this specification.
Interoperability considerations: There are no known interoperability issues.

Published specification: This specification.
Applications that use this media type: This media type is intended primarily for use with the HTTP PATCH method.
Additional information:
Magic number(s): N/A
File extension(s): N/A
Macintosh file type code(s): "TEXT"
Person & email address to contact for further information: James M Snell <jasnell@gmail.com>
Intended usage: COMMON
Restrictions on usage: None.
Author: James M Snell <jasnell@gmail.com>
Change controller: IETF

5. Security Considerations

The "application/merge-patch" Media Type and all Type-Specific Variations are intended for use as a mechanism to allow user-agents requesting partial modification of a target resource to indicate an intention for the server to determine the specific set of change operations that are to be applied to the target resource. As such, it is the servers responsibility to determine the appropriateness of any given change as well as the user-agents authorization to request such changes.

All of the the security considerations discussed in Section 5 [RFC5789] apply to all uses of the HTTP PATCH method with the "application/merge-patch" Media Type (and all Type-Specific Variations).

6. References

6.1. Normative References

[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.

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

6.2. Informational References

[I-D.ietf-appsawg-json-patch]
Bryan, P., "JSON Patch", draft-ietf-appsawg-json-patch-01 (work in progress), March 2012.

Author's Address

James M Snell

Email: jasnell@gmail.com