

# Link Relation Types for Simple Version Navigation between Web Resources

## Abstract

This specification defines a set of link relation types that may be used on Web resources for navigation between a resource and other resources related to version control, such as past versions and working copies.

## Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at <http://www.rfc-editor.org/info/rfc5829><sup>1</sup>.

## Copyright Notice

Copyright © 2010 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><sup>2</sup>) 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.

<sup>1</sup> <http://www.rfc-editor.org/info/rfc5829>

<sup>2</sup> <http://trustee.ietf.org/license-info>

## Table of Contents

<b>1 Introduction</b> .....	<b>3</b>
<b>2 Terminology</b> .....	<b>4</b>
<b>3 Link Relations</b> .....	<b>5</b>
3.1 'version-history'.....	5
3.2 'latest-version'.....	5
3.3 'working-copy'.....	5
3.4 'working-copy-of'.....	5
3.5 'predecessor-version'.....	5
3.6 'successor-version'.....	5
<b>4 IANA Considerations</b> .....	<b>6</b>
4.1 'version-history' Link Relation Registration.....	6
4.2 'latest-version' Link Relation Registration.....	6
4.3 'working-copy' Link Relation Registration.....	6
4.4 'working-copy-of' Link Relation Registration.....	6
4.5 'predecessor-version' Link Relation Registration.....	6
4.6 'successor-version' Link Relation Registration.....	7
<b>5 Security Considerations</b> .....	<b>8</b>
<b>6 Acknowledgments</b> .....	<b>9</b>
<b>7 References</b> .....	<b>10</b>
7.1 Normative References.....	10
7.2 Informative References.....	10
<b>Authors' Addresses</b> .....	<b>11</b>
<b>A Relationship to Java Content Repository (JCR) and WebDAV</b> .....	<b>12</b>
A.1 Example: Use of Link Relations in HTTP Link Header.....	12

## 1. Introduction

This specification defines a set of link relation types that may be used on Web resources that exist in a system that supports versioning to navigate among the different resources available, such as past versions and working copies.

These link relations are used in the AtomPub ([RFC5023](#)) bindings of the "Content Management Interoperability Services" (CMIS). See Section 3.4.3.3 of [\[CMIS\]](#) for further information.

## 2. Terminology

### *Versioned Resource*

When a resource is put under version control, it becomes a "versioned resource". Many servers protect versioned resources from modifications by considering them "checked in", and by requiring a "checkout" operation before modification, and a "checkin" operation to get back to the "checked-in" state. Other servers allow modification, in which case the checkout/checkin operation may happen implicitly.

### *Version History*

A "version history" resource is a resource that contains all the versions of a particular versioned resource.

### *Predecessor, Successor*

When a versioned resource is checked out and then subsequently checked in, the version that was checked out becomes a "predecessor" of the version created by the checkin. A client can specify multiple predecessors for a new version if the new version is logically a merge of those predecessors. The inverse of the predecessor relation is the "successor" relation. Therefore, if X is a predecessor of Y, then Y is a successor of X.

### *Working Copy*

A "working copy" is a resource at a server-defined URL that can be used to create a new version of a versioned resource.

### *Checkout*

A "checkout" is an operation on a versioned resource that creates a working copy, or changes the versioned resource to be a working copy as well ("in-place versioning").

### *Checkin*

A "checkin" is an operation on a working copy that creates a new version of its corresponding versioned resource.

**Note:** the operations for putting a resource under version control and for checking in and checking out depend on the protocol in use and are beyond the scope of this document; see [\[CMIS\]](#), [\[RFC3253\]](#), and [\[JSR-283\]](#) for examples.

### 3. Link Relations

The following link relations are defined.

#### 3.1 'version-history'

When included on a versioned resource, this link points to a resource containing the version history for this resource.

#### 3.2 'latest-version'

When included on a versioned resource, this link points to a resource containing the latest (e.g., current) version.

The latest version is defined by the system. For linear versioning systems, this is probably the latest version by timestamp. For systems that support branching, there will be multiple latest versions, one for each branch in the version history.

Some systems may allow more than one of these link relations.

#### 3.3 'working-copy'

When included on a versioned resource, this link points to a working copy for this resource.

Some systems may allow more than one of these link relations.

#### 3.4 'working-copy-of'

When included on a working copy, this link points to the versioned resource from which this working copy was obtained.

#### 3.5 'predecessor-version'

When included on a versioned resource, this link points to a resource containing the predecessor version in the version history.

Some systems may allow more than one of these link relations in the case of multiple branches merging.

#### 3.6 'successor-version'

When included on a versioned resource, this link points to a resource containing the successor version in the version history.

Some systems may allow more than one of these link relations in order to support branching.

## 4. IANA Considerations

The link relations below have been registered by IANA per Section 7.1 of [\[RFC4287\]](#):

### 4.1 'version-history' Link Relation Registration

~~Attribute~~ version-history

Value:

~~Description:~~ 3.1.

~~Expanded:~~ this relation can be used for background processing or to provide extended functionality without displaying its value.

characteristics:

~~See:~~ [Section 5](#).

considerations:

### 4.2 'latest-version' Link Relation Registration

~~Attribute~~ latest-version

Value:

~~Description:~~ 3.2.

~~Expanded:~~ this relation can be used for background processing or to provide extended functionality without displaying its value.

characteristics:

~~See:~~ [Section 5](#).

considerations:

### 4.3 'working-copy' Link Relation Registration

~~Attribute~~ working-copy

Value:

~~Description:~~ 3.3.

~~Expanded:~~ this relation can be used for background processing or to provide extended functionality without displaying its value.

characteristics:

~~See:~~ [Section 5](#).

considerations:

### 4.4 'working-copy-of' Link Relation Registration

~~Attribute~~ working-copy-of

Value:

~~Description:~~ 3.4.

~~Expanded:~~ this relation can be used for background processing or to provide extended functionality without displaying its value.

characteristics:

~~See:~~ [Section 5](#).

considerations:

### 4.5 'predecessor-version' Link Relation Registration

~~Attribute~~ predecessor-version

Value:

~~See Section 3.5.~~

~~Explicitly defined; this relation can be used for background processing or to provide extended functionality without displaying its value.~~

~~characteristics:~~

~~See Section 5.~~

~~considerations:~~

## 4.6 'successor-version' Link Relation Registration

~~Attribute:~~ successor-version

~~Value:~~

~~See Section 3.6.~~

~~Explicitly defined; this relation can be used for background processing or to provide extended functionality without displaying its value.~~

~~characteristics:~~

~~See Section 5.~~

~~considerations:~~

## 5. Security Considerations

Automated agents should take care when these relations cross administrative domains (e.g., the URI has a different authority than the current document). Such agents should also take care to detect circular references.

Care should be applied when versioned resources are subject to differing access policies. In this case, exposing links may leak information even if the linked resource itself is properly secured. In particular, the syntax of the link target could expose sensitive information (see Section 16.2 of [\[RFC3253\]](#) for a similar consideration in WebDAV Versioning). Note that this applies to exposing link metadata in general, not only to links related to versioning.

## 6. Acknowledgments

Thanks to the members of Content Management Interoperability Services (CMIS) Technical Committee (TC) at OASIS for the initial proposal, and to Jan Algermissen for feedback during IETF review.

## 7. References

### 7.1 Normative References

- [RFC4287] Nottingham, M., Ed. and R. Sayre, Ed., "[The Atom Syndication Format](#)", RFC 4287, December 2005.

### 7.2 Informative References

- [CMIS] Brown, A., Gur-Esh, E., McVeigh, R., and F. Mueller, "[Content Management Interoperability Services \(CMIS\) Version 1.0](#)", OASIS Content Management Interoperability Services (CMIS) Version 1.0 Committee Specification 01, March 2010, <<http://docs.oasis-open.org/cmisis/CMIS/v1.0/cs01/cmisis-spec-v1.0.html>>. Latest version available at <<http://docs.oasis-open.org/cmisis/CMIS/v1.0/cmisis-spec-v1.0.html>>
- [JSR-283] Day Software, Nuescheler, D., and P. Piegaze, "[Content Repository API for Java\(tm\) Technology Specification](#)", Java Specification Request 283, August 2009, <<http://www.day.com/specs/jcr/2.0/>>.
- [RFC3253] Clemm, G., Amsden, J., Ellison, T., Kaler, C., and J. Whitehead, "[Versioning Extensions to WebDAV \(Web Distributed Authoring and Versioning\)](#)", RFC 3253, March 2002.
- [RFC5023] Gregorio, J. and B. de hOra, "[The Atom Publishing Protocol](#)", RFC 5023, October 2007.
- [WEB-LINKING] Nottingham, M., "Web Linking", Work in Progress, March 2010.

## Authors' Addresses

**Al Brown**

IBM

3565 Harbor Blvd

Costa Mesa, California 92626

USA

EMail: [albertcbrown@us.ibm.com](mailto:albertcbrown@us.ibm.com)**Geoffrey Clemm**

IBM

20 Maguire Road

Lexington, MA 02421

USA

EMail: [geoffrey.clemm@us.ibm.com](mailto:geoffrey.clemm@us.ibm.com)**Julian F. Reschke** (editor)

greenbytes GmbH

Hafenweg 16

Muenster, NW 48155

Germany

EMail: [julian.reschke@greenbytes.de](mailto:julian.reschke@greenbytes.de)URI: <http://greenbytes.de/tech/webdav/>

## A. Relationship to Java Content Repository (JCR) and WebDAV

The link relations defined in [Section 3](#) correspond to various properties used in WebDAV Versioning [\[RFC3253\]](#) and JCR [\[JSR-283\]](#):

### version-history

WebDAV: the resource identified by the DAV:version-history property ([\[RFC3253\]](#), Sections 5.2.1 and 5.3.1).

JCR: the node identified by jcr:versionHistory property ([\[JSR-283\]](#), Section 3.13.2.4) for versionable nodes, the parent folder for version nodes.

### latest-version

WebDAV: for version-controlled resources, DAV:checked-in ([\[RFC3253\]](#), Section 3.2.1) or DAV:checked-out ([\[RFC3253\]](#), Section 3.3.1), depending on checkin state. For version resources, a successor version that itself does not have any successors.

JCR: the version node identified by the jcr:baseVersion property ([\[JSR-283\]](#), Section 3.13.2.5) for versionable nodes; for version nodes, a successor version that itself does not have any successors.

### working-copy

WebDAV: for version-controlled resources that are checked-out in place: the resource itself. For version resources: each resource identified by a member of the DAV:checkout-set property (see [\[RFC3253\]](#), Section 3.4.3).

JCR: for checked-out versionable nodes: the node itself.

### working-copy-of

WebDAV: the resource identified by the DAV:checked-out property (see [\[RFC3253\]](#), Section 3.3.1).

JCR: for checked-out versionable nodes: the node identified by the jcr:baseVersion property ([\[JSR-283\]](#), Section 3.13.12.5).

### predecessor-version

WebDAV: each resource identified by a member of DAV:predecessor-set ([\[RFC3253\]](#), Sections 3.3.2 and 3.4.1).

JCR: each node identified by a member of jcr:predecessors ([\[JSR-283\]](#), Section 3.13.3.3).

### successor-version

WebDAV: each resource identified by a member of DAV:successor-set ([\[RFC3253\]](#), Section 3.4.2).

JCR: each node identified by a member of jcr:successors ([\[JSR-283\]](#), Section 3.13.3.4).

### A.1 Example: Use of Link Relations in HTTP Link Header

The "Web Linking" specification ([\[WEB-LINKING\]](#)) generalizes Atom link relations, and also reintroduces the HTTP "Link" header as a way to expose link relations in HTTP responses. This will make it possible to expose version links independently from a specific vocabulary, be it the Atom Feed Format ([\[RFC4287\]](#)) or WebDAV properties ([\[RFC3253\]](#)).

For instance, a response to a VERSION-CONTROL request ([\[RFC3253\]](#), Section 3.5) could expose a newly created version-history and checked-in version as link relations:

>> Request:

```
VERSION-CONTROL /docs/test.txt HTTP/1.1
Host: example.net
```

>> Response:

```
HTTP/1.1 204 No Content
Link: </system/v/84345634/1>; rel=latest-version;
      anchor=</docs/test.txt>
Link: </system/vh/84345634>; rel=version-history;
      anchor=</docs/test.txt>
```

(Note that in this case, the anchor parameter is used, as the response to a VERSION-CONTROL request is not a representation of the resource at the Request-URI.)

A subsequent HEAD request on that resource could expose the version-history and latest-version relations as well:

>> Request:

```
HEAD /docs/test.txt HTTP/1.1
Host: example.net
```

>> Response:

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=UTF-8
Content-Length: 12345
Link: </system/v/84345634/1>; rel=latest-version
Link: </system/vh/84345634>; rel=version-history
```

After creating more versions, following the latest-version would then expose predecessors of a version:

>> Request:

```
HEAD /system/v/84345634/3 HTTP/1.1
Host: example.net
```

>> Response:

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=UTF-8
Content-Length: 12323
Link: </system/v/84345634/2>; rel=predecessor-version
```