

WS-Calendar Calendar Update and Synchronization with REST-based Services Version 1.0

Committee Specification Draft 02

15 February 2013

Specification URIs

This version:

<http://docs.oasis-open.org/ws-calendar/ws-calendar-rest/v1.0/csd02/ws-calendar-rest-v1.0-csd02.pdf> (Authoritative)
<http://docs.oasis-open.org/ws-calendar/ws-calendar-rest/v1.0/csd02/ws-calendar-rest-v1.0-csd02.html>
<http://docs.oasis-open.org/ws-calendar/ws-calendar-rest/v1.0/csd02/ws-calendar-rest-v1.0-csd02.doc>

Previous version:

<http://docs.oasis-open.org/ws-calendar/ws-calendar-rest/v1.0/csprd01/ws-calendar-rest-v1.0-csprd01.pdf> (Authoritative)
<http://docs.oasis-open.org/ws-calendar/ws-calendar-rest/v1.0/csprd01/ws-calendar-rest-v1.0-csprd01.html>
<http://docs.oasis-open.org/ws-calendar/ws-calendar-rest/v1.0/csprd01/ws-calendar-rest-v1.0-csprd01.doc>

Latest version:

<http://docs.oasis-open.org/ws-calendar/ws-calendar-rest/v1.0/ws-calendar-rest-v1.0.pdf> (Authoritative)
<http://docs.oasis-open.org/ws-calendar/ws-calendar-rest/v1.0/ws-calendar-rest-v1.0.html>
<http://docs.oasis-open.org/ws-calendar/ws-calendar-rest/v1.0/ws-calendar-rest-v1.0.doc>

Technical Committee:

OASIS Web Services Calendar (WS-Calendar) TC

Chair:

Toby Considine (toby.considine@unc.edu), University of North Carolina at Chapel Hill

Editor:

Michael Douglass (douglm@rpi.edu) Rensselaer Polytechnic Institute

Related work:

This specification is related to:

- *WS-Calendar Version 1.0*. Latest version.
<http://docs.oasis-open.org/ws-calendar/ws-calendar/v1.0/ws-calendar-1.0-spec.html>

Abstract:

This document describes standard messages and interactions for update and synchronization using REST with a system that hosts calendar-based information. Hosted information can be either traditional personal and enterprise calendar information or services that support XML payloads developed in conformance with the WS-Calendar specification.

Status:

This document was last revised or approved by the OASIS Web Services Calendar (WS-Calendar) TC on the above date. The level of approval is also listed above. Check the “Latest version” location noted above for possible later revisions of this document.

Technical Committee members should send comments on this specification to the Technical Committee’s email list. Others should send comments to the Technical Committee by using the “Send A Comment” button on the Technical Committee’s web page at <http://www.oasis-open.org/committees/ws-calendar/> .

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (<http://www.oasis-open.org/committees/ws-calendar/ipr.php>).

Citation format:

When referencing this specification the following citation format should be used:

[WS-Calendar-REST]

WS-Calendar Calendar Update and Synchronization with REST-based Services Version 1.0. 15 February 2013. OASIS Committee Specification Draft 02. <http://docs.oasis-open.org/ws-calendar/ws-calendar-rest/v1.0/csd02/ws-calendar-rest-v1.0-csd02.html>.

Notices

Copyright © OASIS Open 2013. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full [Policy](#) may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name "OASIS" is a trademark of [OASIS](#), the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see <http://www.oasis-open.org/policies-guidelines/trademark> for above guidance.

Table of Contents

1	Introduction.....	7
1.1	Terminology.....	7
1.2	Normative References.....	7
1.3	Non-Normative References.....	7
1.4	Namespace.....	7
2	Calendar Services.....	8
2.1	Overview of the protocol.....	8
2.1.1	Calendar Object Resources.....	8
2.1.2	Timezone information.....	8
2.1.3	Issues not addressed by this specification.....	9
2.1.4	CalWS Glossary.....	9
2.2	Error conditions.....	10
2.2.1	Example: error with CalDAV error condition.....	10
3	Properties and link relations.....	11
3.1	Property and relation-type URIs.....	11
3.2	supported-features property.....	11
3.3	max-attendees-per-instance.....	11
3.4	max-date-time.....	11
3.5	max-instances.....	11
3.6	max-resource-size.....	11
3.7	min-date-time.....	12
3.8	description.....	12
3.9	timezone-service relation.....	12
3.10	principal-home relation.....	12
3.11	current-principal-freebusy relation.....	12
3.12	principal-freebusy relation.....	12
3.13	child-collection relation.....	12
3.14	created link property.....	13
3.15	last-modified property.....	13
3.16	displayname property.....	13
3.17	timezone property.....	13
3.18	owner property.....	13
3.19	collection link property.....	13
3.20	calendar-collection link property.....	13
3.21	calWS:privilege-set XML element.....	14
4	Retrieving Collection and Service Properties.....	15
4.1	Request parameters.....	15
4.2	Responses:.....	15
4.3	Example - retrieving server properties:.....	15
5	Creating Calendar Object Resources.....	17
5.1	Request parameters.....	17
5.2	Responses:.....	17
5.3	Preconditions for Calendar Object Creation.....	17

5.4 Example - successful POST:	18
5.5 Example - unsuccessful POST:	18
6 Retrieving resources	19
6.1 Request parameters	19
6.2 Responses:	19
6.3 Example - successful fetch:	19
6.4 Example - unsuccessful fetch:	19
7 Updating resources	20
7.1 Responses:	20
8 Deletion of resources	22
8.1 Delete for Collections	22
8.2 Responses:	22
9 Querying calendar resources	23
9.1 Limiting data returned	23
9.2 Pre/postconditions for calendar queries	23
9.3 Example: time range limited retrieval	23
10 Free-busy queries	28
10.1 ACCEPT header	28
10.2 URL Query Parameters	28
10.2.1 start	28
10.2.2 end	29
10.2.3 period	29
10.2.4 account	29
10.3 URL parameters - notes	29
10.4 HTTP Operations	29
10.5 Response Codes	29
10.6 Examples	30
11 Conformance	33
11.1 Start, end and duration in calendar components	33
11.1.1 Updating, transporting and maintaining start, and and duration.	33
11.1.2 VEVENT:	33
11.1.3 VTODO:	33
11.1.4 VJOURNAL:	34
11.1.5 VAVAILABILITY	34
11.1.6 AVAILABILITY	34
11.2 Recurrences.	34
11.3 Alarms:	34
11.4 Unrecognized or unsupported elements	34
Appendix A. Acknowledgments	35
Appendix B. An Introduction to Internet Calendaring	36
B.1 icalendar	36
B.1.1 History	36
B.1.2 Data model	36
B.1.3 Scheduling	37
B.1.4 Extensibility	37

B.2 Calendar data access and exchange protocols	37
B.2.1 Internet Calendar Subscriptions.....	37
B.2.2 CalDAV	37
B.2.3 ActiveSync/SyncML	38
B.2.4 CalWS.....	38
B.2.5 iSchedule	38
B.3 References	38
Appendix C. Revision History	39

1 Introduction

The CalWS REST protocol is built upon and makes the same assumptions about structure as the CalDAV protocol defined in [RFC 4791] and related specifications. It does NOT require nor assume the WebDAV nor CalDAV protocol.

Calendar resources, for example events and tasks are stored as named resources (files) inside special collections (folders) known as "**Calendar Collections**".

This specification can be looked upon as a layer built on top of CalDAV and defines the basic operations which allow creation, retrieval, update and deletion. In addition, query and freebusy operations are defined to allow efficient, partial retrieval of calendar data.

This does not mean that a CalWS service must be built on CalDAV, merely that a degree of conformity is established such that services built in that manner do not have a significant mismatch. It is assumed that some CalWS REST services will be built without any CalDAV support.

1.1 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 [RFC2119].

1.2 Normative References

- [RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.
- [RFC4791] C. Daboo, B. Desruisseaux, L. Dusseault, *Calendar Extensions to WebDAV (CalDAV)*, <http://www.ietf.org/rfc/rfc4791.txt>, IETF RFC4791, March 1997.
- [WS-Calendar] *WS-Calendar Version 1.0*. 19 January 2011. OASIS Committee Specification <http://docs.oasis-open.org/ws-calendar/ws-calendar-spec/v1.0/cs01/ws-calendar-spec-v1.0-cs01.pdf>.
- [XRD] Extensible Resource Descriptor (XRD) Version 1.0, 1 November 2010, OASIS Standard, <http://docs.oasis-open.org/xri/xrd/v1.0/os/xrd-1.0-os.xml>

1.3 Non-Normative References

- REST T Fielding, *Architectural Styles and the Design of Network-based Software Architectures*, <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>.

1.4 Namespace

XML namespaces and prefixes used in this standard:

Table 1 1: XML Namespaces in this standard

Prefix	Namespace
xcal	urn:ietf:params:xml:ns:icalendar-2.0
calWS	http://docs.oasis-open.org/ws-calendar/ns/REST
xrd	http://docs.oasis-open.org/ns/xri/xrd-1.0

2 Calendar Services

36 The Service interactions are built upon and make the same assumptions about structure as the CalDAV
37 protocol defined in **[RFC4791]** and related specifications. It does NOT require nor assume the WebDAV
38 nor CalDAV protocol but does make use of some of the same elements and structures in the CalDAV
39 XML namespace.

40 Calendar resources, for example events and tasks are stored as named resources (files) inside special
41 collections (folders) known as "**Calendar Collections**".

42 These services can be looked upon as a layer built on top of CalDAV and defines the basic operations
43 which allow creation, retrieval, update and deletion. In addition, query, and free-busy operations are
44 defined to allow efficient, partial retrieval of calendar data.

45 These services assume a degree of conformity with CalDAV is established such that services built in that
46 manner do not have a significant mismatch. It is assumed that some WS-Calendar services will be built
47 without any CalDAV support.

2.1 Overview of the protocol

48 The protocol is an HTTP based RESTfull protocol using a limited set of methods. Each request may be
49 followed by a response containing status information.

51 The following methods are specified in the protocol description, PUT, POST, GET, DELETE. To avoid
52 various issues with certain methods being blocked clients may use the X-HTTP-Method-Override: header
53 to specify the intended operation. Servers SHOULD behave as if the named method was used.

```
54 POST /user/fred/calendar/ HTTP/1.1  
55 ...  
56 X-HTTP-Method-Override: PUT  
57 Properties
```

58 A service or resource will have a number of properties which describe the current state of that service or
59 resource. These properties are accessed through a GET on the target resource or service with an
60 ACCEPT header specifying application/xrd+xml. See Section 2.1.3.6

61 The following operations are defined by this specification:

- 62 • Retrieval and update of service and resource properties
- 63 • Creation of a calendar object
- 64 • Retrieval of a calendar object
- 65 • Update of a calendar object
- 66 • Deletion of a calendar object
- 67 • Query
- 68 • Free-busy query

2.1.1 Calendar Object Resources

70 The same restrictions apply to Calendar Object Resources as specified in CalDAV **[RFC4791]** section
71 4.2. An additional constraint for CalWS is that no timezone specifications are transferred.

2.1.2 Timezone information

73 It is assumed that the client and server each have access to a full set of up to date timezone information.
74 Timezones will be referenced by a timezone identifier from the full set of Olson data together with a set of
75 well-known aliases defined **[TZDB]**. CalWS services may advertise themselves as timezone servers
76 through the server properties object.

77 **2.1.3 Issues not addressed by this specification.**

78 A number of issues are not addressed by this version of the specification, either because they should be
79 addressed elsewhere or will be addressed at some later date.

80 **2.1.3.1 Access Control**

81 It is assumed that the targeted server will set an appropriate level of access based on authentication. This
82 specification will not attempt to address the issues of sharing or Access Control Lists (ACLs).

83 **2.1.3.2 Provisioning**

84 The protocol will not provide any explicit provisioning operations. If it is possible to authenticate or
85 address a principal's calendar resources then they **MUST** be automatically created if necessary or
86 appropriate

87 **2.1.3.3 Copy/Move**

88 These operations are not yet defined for this version of the CalWS protocol. Both operations raise a
89 number of issues. In particular implementing a move operation through a series of retrievals, insertions
90 and deletions may cause undesirable side-effects. Both these operations will be defined in a later version
91 of this specification.

92 **2.1.3.4 Creating Collections**

93 We will not address the issue of creating collections within the address space. The initial set is created by
94 provisioning.

95 **2.1.3.5 Retrieving collections**

96 This operation is currently undefined. A GET on a collection may fail or return a complete calendar object
97 representing the collection.

98 **2.1.3.6 Setting service and resource properties.**

99 These operations are not defined in this version of the specification. In the future it will be possible to
100 define or set the properties for the service or resources within the service.

101 **2.1.4 CalWS Glossary**

102 **2.1.4.1 Hrefs**

103 An href is a URI reference to a resource, for example

104 `"http://example.org/user/fred/calendar/event1.ics".`

105 The URL above reflects a possible structure for a calendar server. All URLs should be absolute or path-
106 absolute following the rules defined in Section 3.1: Property and relation-type URIs

107 **2.1.4.2 Calendar Object Resource**

108 A calendar object resource is an event, meeting or a task. Attachments are resources but NOT calendar
109 object resources. An event or task with overrides is a single calendar resource entity.

110 **2.1.4.3 Calendar Collection**

111 A folder only allowed to contain calendar object resources.

112 2.1.4.4 Scheduling Calendar Collection

113 A folder only allowed to contain calendar resources which is also used for scheduling operations.
114 Scheduling events placed in such a collection will trigger implicit scheduling activity on the server.

115 2.1.4.5 Principal Home

116 The collection under which all the resources for a given principal are stored. For example, for principal
117 "fred" the principal home might be "/user/fred/"

118 2.2 Error conditions

119 Each operation on the calendar system has a number of pre-conditions and post-conditions that apply.
120 A "precondition" for a method describes the state of the server that must be true for that method to be
121 performed. A "post-condition" of a method describes the state of the server that must be true after that
122 method has been completed. Any violation of these conditions will result in an error response in the form
123 of a CalWS XML error element containing the violated condition and an optional description. \
124 Each method specification defines the preconditions that must be satisfied before the method can
125 succeed. A number of post-conditions are generally specified which define the state that must exist after
126 the execution of the operation. Preconditions and post-conditions are defined as error elements in the
127 CalWS XML namespace.

128 2.2.1 Example: error with CalDAV error condition

```
129 <?xml version="1.0" encoding="utf-8"  
130     xmlns:CW="Error! Reference source not found."  
131     xmlns:C="http://docs.oasis-open.org/ws-calendar/ns/REST" ?>  
132 <CW:error>  
133   <C:supported-filter>  
134     <C:prop-filter name="X-ABC-GUID"/>  
135   </C:supported-filter>  
136   <CW:description>Unknown property </CW:description>  
137 </CW:error>
```

138

3 Properties and link relations

139

3.1 Property and relation-type URIs

140

In the XRD entity returned properties and related services and entities are defined by absolute URIs which correspond to the extended relation type defined in **[web linking]** Section 4.2. These URIs do NOT correspond to any real entity on the server and clients should not attempt to retrieve any data at that target.

141

142

143

144

Certain of these property URIs correspond to CalDAV preconditions. Each URL is prefixed by the CalWS relations and properties namespace `http://docs.oasis-open.org/ws-calendar/ns/REST/`. Those properties which correspond to CalDAV properties have the additional path element "caldav/", for example

145

146

147

```
http://docs.oasis-open.org/ws-calendar/ns/REST/supported-calendar-data
```

148

corresponds to

149

```
CalDAV:supported-calendar-data
```

150

In addition to those CalDAV properties, the CalWS specification defines a number of other properties and link relations with the URI prefix of `http://docs.oasis-open.org/ws-calendar/ns/REST/`.

151

152

3.2 supported-features property.

153

`http://docs.oasis-open.org/ws-calendar/ns/REST/supported-features`

154

This property defines the features supported by the target. All resources contained and managed by the service should return this property. The value is a comma separated list containing one or more of the following

155

156

157

- calendar-access - the service supports all MUST requirements in this specification

158

159

160

```
<Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/supported-features"
>calendar-access</Property>
```

161

3.3 max-attendees-per-instance

162

`http://docs.oasis-open.org/ws-calendar/ns/REST/max-attendees-per-instance`

163

Defines the maximum number of attendees allowed per event or task.

164

3.4 max-date-time

165

`http://docs.oasis-open.org/ws-calendar/ns/REST/max-date-time`

166

Defines the maximum date/time allowed on an event or task

167

3.5 max-instances

168

`http://docs.oasis-open.org/ws-calendar/ns/REST/max-instances`

169

Defines the maximum number of instances allowed per event or task

170

3.6 max-resource-size

171

`http://docs.oasis-open.org/ws-calendar/ns/REST/max-resource-size`

172

Provides a numeric value indicating the maximum size of a resource in octets that the server is willing to accept when a calendar object resource is stored in a calendar collection.

173

174 3.7 min-date-time

175 <http://docs.oasis-open.org/ws-calendar/ns/REST/min-date-time>

176 Provides a DATE-TIME value indicating the earliest date and time (in UTC) that the server is willing to
177 accept for any DATE or DATE-TIME value in a calendar object resource stored in a calendar collection.

178 3.8 description

179 <http://docs.oasis-open.org/ws-calendar/ns/REST/description>

180 Provides some descriptive text for the targeted collection.

181 3.9 timezone-service relation.

182 <http://docs.oasis-open.org/ws-calendar/ns/REST/timezone-service>

183 The location of a timezone service used to retrieve timezone information and specifications. This may be
184 an absolute URL referencing some other service or a relative URL if the current server also provides a
185 timezone service.

```
186 <Link rel="http://docs.oasis-open.org/ws-calendar/ns/REST/timezone-service"  
187 href="http://example.com/tz" />
```

188 3.10 principal-home relation.

189 <http://docs.oasis-open.org/ws-calendar/ns/REST/principal-home>

190 Provides the URL to the user home for the currently authenticated principal.

```
191 <Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/principal-home"  
192 href="http://example.com/user/fred" />
```

193 3.11 current-principal-freebusy relation.

194 <http://docs.oasis-open.org/ws-calendar/ns/REST/current-principal-freebusy>

195 Provides the URL to use as a target for freebusy requests for the current authenticated principal.

```
196 <Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/current-principal-freebusy"  
197 href="http://example.com/freebusy/user/fred" />
```

198 3.12 principal-freebusy relation.

199 <http://docs.oasis-open.org/ws-calendar/ns/REST/principal-freebusy>

200 Provides the URL to use as a target for freebusy requests for a different principal.

```
201 <Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/principal-freebusy"  
202 href="http://example.com/freebusy" />
```

203 3.13 child-collection relation.

204 <http://docs.oasis-open.org/ws-calendar/ns/REST/child-collection>

205 Provides information about a child collections for the target. The href attribute gives the URI of the
206 collection. The element should only have CalWS child elements giving the type of the collection, that is
207 the calWS:collection link property and the CalWS-calendar-collection link property. This allows clients to
208 determine the structure of a hierarchical system by targeting each of the child collections in turn.

209 The xrd:title child element of the link element provides a description for the child-collection.

```
210 <Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/child-collection"  
211 href="http://example.com/calWS/user/fred/calendar">  
212 <Title xml:lang="en">Calendar</Title>  
213 <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/collection"  
214 xsi:nil="true" />
```

```
215 <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/calendar-
216 collection"
217 xsi:nil="true" />
218 </Link>
```

219 3.14 created link property

220 <http://docs.oasis-open.org/ws-calendar/ns/REST/created>

221 Appears within a link relation describing collections or entities. The value is a date-time as defined in
222 **[WS-Calendar]** Section 5.6

```
223 <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/created"
224 >1985-04-12T23:20:50.52Z</Property>
```

225 3.15 last-modified property

226 <http://docs.oasis-open.org/ws-calendar/ns/REST/last-modified>

227 Appears within an xrd object describing collections or entities. The value is the same format as would
228 appear in the Last-Modified header and is defined in **[RFC2616]**, Section 3.3.1

```
229 <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/last-modified"
230 >Mon, 12 Jan 1998 09:25:56 GMT</Property>
```

231 3.16 displayname property

232 <http://docs.oasis-open.org/ws-calendar/ns/REST/displayname>

233 Appears within an xrd object describing collections or entities. The value is a localized name for the entity
234 or collection.

```
235 <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/displayname"
236 >My Calendar</Property>
```

237 3.17 timezone property

238 <http://docs.oasis-open.org/ws-calendar/ns/REST/timezone>

239 Appears within an xrd object describing collections. The value is a text timezone identifier.

```
240 <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/timezone"
241 >America/New_York</Property>
```

242 3.18 owner property

243 <http://docs.oasis-open.org/ws-calendar/ns/REST/owner>

244 Appears within an xrd object describing collections or entities. The value is a server specific uri.

```
245 <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/owner"
246 >/principals/users/mike</Property>
```

247 3.19 collection link property

248 <http://docs.oasis-open.org/ws-calendar/ns/REST/collection>

249 Appears within a link relation describing collections or entities. The property takes no value and indicates
250 that this child element is a collection.

```
251 <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/collection"
252 xsi:nil="true" />
```

253 3.20 calendar-collection link property

254 <http://docs.oasis-open.org/ws-calendar/ns/REST/calendar-collection>

255 Appears within a link relation describing collections or entities. The property takes no value and indicates
256 that this child element is a calendar collection.

```
257 <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/calendar-  
258 collection"  
259 xsi:nil="true" />
```

260 3.21 calWS:privilege-set XML element

261 <http://docs.oasis-open.org/ws-calendar/ns/REST/privilege-set>

262 Appears within a link relation describing collections or entities and specifies the set of privileges allowed
263 to the current authenticated principal for that collection or entity.

```
264 <!ELEMENT calWS:privilege-set (calWS:privilege*)>  
265 <!ELEMENT calWS:privilege ANY>
```

266 Each privilege element defines a privilege or access right. The following set is currently defined

- 267 • calWS: Read - current principal has read access
- 268 • calWS: Write - current principal has write access

```
269 <calWS:privilege-set>  
270 <calWS:privilege><calWS:read></calWS:privilege>  
271 <calWS:privilege><calWS:write></calWS:privilege>  
272 </calWS:privilege-set>
```

273

4 Retrieving Collection and Service Properties

274 Properties, related services and locations are obtained from the service or from service resources in the
275 form of an XRD document as defined by [XRD-1.0].

276 Given the URL of a CalWS service a client retrieves the service XRD document through a GET on the
277 service URL with an ACCEPT header specifying application/xrd+xml.

278 Retrieving resource properties is identical to obtaining service properties, that is, execute a GET on the
279 target URL with an ACCEPT header specifying application/xrd+xml.

280 The service properties define the global limits and defaults. Any properties defined on collections within
281 the service hierarchy override those service defaults. The service may choose to prevent such overriding
282 of defaults and limits when appropriate.

283 4.1 Request parameters

- 284 • None

285 4.2 Responses:

- 286 • 200: OK
- 287 • 403: Forbidden
- 288 • 404: Not found

289 4.3 Example - retrieving server properties:

```
290 >>Request
291
292 GET / HTTP/1.1
293 Host: example.com
294 ACCEPT:application/xrd+xml
295
296 >>Response
297 <XRD xmlns="http://docs.oasis-open.org/ns/xri/xrd-1.0"
298     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
299   <Expires>1970-01-01T00:00:00Z</Expires>
300   <Subject>http://example.com/calWS</Subject>
301   <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/created"
302     >1970-01-01</Property>
303
304   <Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/timezone-
305 service"
306     href="http://example.com/tz" />
307
308   <calWS:privilege-set>
309   <calWS:privilege><calWS:read></calWS:privilege>
310   </calWS:privilege-set>
311
312   <Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/principal-home"
313     type="collection"
314     href="http://example.com/calWS/user/fred">
315   <Title xml:lang="en">Fred's calendar home</Title>
316   </Link>
317
318   <Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/child-
319 collection"
320     type="calendar,scheduling"
321     href="http://example.com/calWS/user/fred/calendar">
322   <Title xml:lang="en">Calendar</Title>
```

323
324
325
326
327
328
329
330
331
332
333
334

```
</Link>
  <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/max-
instances"
    >1000</Property>
  <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/max-
attendees-per-instance"
    >100</Property>
</XRD>
```


335
336
337
338

339
340

341
342
343

344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379

5 Creating Calendar Object Resources

Creating calendar object resources is carried out by a POST on the parent collection. The body of the request will contain the resource being created. The request parameter "action=create" indicates this POST is a create. The location header of the response gives the URL of the newly created object.

5.1 Request parameters

- action=create

5.2 Responses:

- 201: created
- 403: Forbidden - no access

5.3 Preconditions for Calendar Object Creation

- **calWS:target-exists:** The target of a PUT must exist. Use POST to create entities and PUT to update them.
- **calWS:not-calendar-data:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST be a supported media type (i.e., iCalendar) for calendar object resources;
- **calWS:invalid-calendar-data:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST be valid data for the media type being specified (i.e., MUST contain valid iCalendar data);
- **calWS:invalid-calendar-object-resource:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST obey all restrictions specified in Calendar Object Resources (e.g., calendar object resources MUST NOT contain more than one type of calendar component, calendar object resources MUST NOT specify the iCalendar METHOD property, etc.);
- **calWS:unsupported-calendar-component:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST contain a type of calendar component that is supported in the targeted calendar collection;
- **calWS:uid-conflict:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST NOT specify an iCalendar UID property value already in use in the targeted calendar collection or overwrite an existing calendar object resource with one that has a different UID property value. Servers SHOULD report the URL of the resource that is already making use of the same UID property value in the calWS:href element
<!ELEMENT uid-conflict (calWS:href)>
- **calWS:invalid-calendar-collection-location:** In a COPY or MOVE request, when the Request-URI is a calendar collection, the Destination-URI MUST identify a location where a calendar collection can be created;
- **calWS:exceeds-max-resource-size:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST have an octet size less than or equal to the value of the CalDAV:max-resource-size property value on the calendar collection where the resource will be stored;
- **calWS:before-min-date-time:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST have all of its iCalendar DATE or DATE-TIME property values (for each recurring instance) greater than or equal to the value of the CalDAV:min-date-time property value on the calendar collection where the resource will be stored;
- **calWS:after-max-date-time:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST have all of its iCalendar DATE or DATE-TIME property values (for each recurring instance) less than the value of the CalDAV:max-date-time property value on the calendar collection where the resource will be stored;

- 380
- **calWS:too-many-instances:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST generate a number of recurring instances less than or equal to the value of the CalDAV: max-instances property value on the calendar collection where the resource will be stored;
- 381
- **calWS:too-many-attendees-per-instance:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST have a number of ATTENDEE properties on any one instance less than or equal to the value of the CalDAV:max-attendees-per-instance property value on the calendar collection where the resource will be stored;
- 382
- 383
- 384
- 385
- 386
- 387

388 5.4 Example - successful POST:

```
389 >>Request
390
391 POST /user/fred/calendar/?action=create HTTP/1.1
392 Host: example.com
393 Content-Type: application/xml+calendar; charset="utf-8"
394 Content-Length: ?
395
396 <?xml version="1.0" encoding="utf-8" ?>
397 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
398   <vcalendar>
399     ...
400   </vcalendar>
401 </icalendar>
402
403 >>Response
404
405 HTTP/1.1 201 Created
406 Location: http://example.com/user/fred/calendar/event1.ics
```

407 5.5 Example - unsuccessful POST:

```
408 >>Request
409
410 POST /user/fred/readcalendar/?action=create HTTP/1.1
411 Host: example.com
412 Content-Type: text/text; charset="utf-8"
413 Content-Length: ?
414
415 This is not an xml calendar object
416
417 >>Response
418
419 HTTP/1.1 403 Forbidden
420 <?xml version="1.0" encoding="utf-8"
421   xmlns:D="DAV:"
422   xmlns:C="urn:ietf:params:xml:ns:caldav" ?>
423 <D:error>
424   <C:supported-calendar-data/>
425   <D:description>Not an icalendar object</D:description>
426 </D:error>
```

427 6 Retrieving resources

428 A simple GET on the href will return a named resource. If that resource is a recurring event or task with
429 overrides, the entire set will be returned. The desired format is specified in the ACCEPT header. The
430 default form is application/xml+calendar

431 6.1 Request parameters

- 432 • none

433 6.2 Responses:

- 434 • 200: OK
- 435 • 403: Forbidden - no access
- 436 • 406 The requested format specified in the accept header is not supported.

437 6.3 Example - successful fetch:

```
438 >>Request
439
440 GET /user/fred/calendar/event1.ics HTTP/1.1
441 Host: example.com
442
443 >>Response
444
445 HTTP/1.1 200 OK
446 Content-Type: application/xml+calendar; charset="utf-8"
447 Content-Length: ?
448
449 <?xml version="1.0" encoding="utf-8" ?>
450 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
451   <vcalendar>
452     ...
453   </vcalendar>
454 </icalendar>
```

455 6.4 Example - unsuccessful fetch:

```
456 >>Request
457
458 PUT /user/fred/calendar/noevent1.ics HTTP/1.1
459 Host: example.com
460
461 >>Response
462
463 HTTP/1.1 404 Not found
```

464

7 Updating resources

465 Resources are updated with the PUT method targeted at the resource href. The body of the request
466 contains a complete new resource which effectively replaces the targeted resource. To allow for optimistic
467 locking of the resource use the if-match header.

468 When updating a recurring event all overrides and master must be supplied as part of the content.

469 Preconditions as specified in Section 5.3 are applicable.

470 7.1 Responses:

- 471 • 200: OK
- 472 • 304: Not modified - entity was modified by some other request
- 473 • 403: Forbidden - no access, does not exist etc. See error response
- 474 •

475 *Example 7-1: Successful Update*

```
476 >>Request
477
478 PUT /user/fred/calendar/event1.ics HTTP/1.1
479 Host: example.com
480 Content-Type: application/xml+calendar; charset="utf-8"
481 Content-Length: ?
482
483 <?xml version="1.0" encoding="utf-8" ?>
484 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
485   <vcalendar>
486     ...
487   </vcalendar>
488 </icalendar>
489
490 >>Response
491
492 HTTP/1.1 200 OK
```

493 *Example 7-2: Unsuccessful Update*

```
494 >>Request
495
496 PUT /user/fred/readcalendar/event1.ics HTTP/1.1
497 Host: example.com
498 Content-Type: application/xml+calendar; charset="utf-8"
499 Content-Length: ?
500
501 <?xml version="1.0" encoding="utf-8" ?>
502 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
503   <vcalendar>
504     ...
505   </vcalendar>
506 </icalendar>
507
508 >>Response
509
510 HTTP/1.1 403 Forbidden
511 Content-Type: application/xml; charset="utf-8"
512 Content-Length: xxxx
513
514 <?xml version="1.0" encoding="utf-8"
515     xmlns:D="DAV:"
516     xmlns:CW=" http://docs.oasis-open.org/ws-calendar/ns/REST/" ?>
```

```
517 <CW:error>  
518   <CW:target-exists/>  
519   <CW:description>Target of update must exist</C:description>  
520 </CW:error>
```

521 **8 Deletion of resources**

522 Delete is defined in **[RFC 2616]** Section 9.7. In addition to conditions defined in that specification, servers
523 must remove any references from the deleted resource to other resources. Resources are deleted with
524 the DELETE method targeted at the resource URL. After a successful completion of a deletion a GET on
525 that URL must result in a 404 - Not Found status.

526 **8.1 Delete for Collections**

527 Delete for collections may or may not be supported by the server. Certain collections are considered
528 undeletable. On a successful deletion of a collection all contained resources to any depth must also be
529 deleted.

530 **8.2 Responses:**

- 531 • 200: OK
532 • 403: Forbidden - no access
533 • 404: Not Found

534 9 Querying calendar resources

535 Querying provides a mechanism by which information can be obtained from the service through possibly
536 complex queries. A list of icalendar properties can be specified to limit the amount of information returned
537 to the client. A query takes the parts

- 538 • Limitations on the data returned
- 539 • Selection of the data
- 540 • Optional timezone id for floating time calculations.

541 The current specification uses CalDAV multiget and calendar-query XML bodies as specified in **[RFC**
542 **4791]** with certain limitations and differences.

- 543 1. The POST method is used for all requests, the action being identified by the outer element.
- 544 2. While CalDAV servers generally only support [RFC 5545] and assume that as the default, the
545 delivery format for CalWS will, by default, be [draft-xcal].
- 546 3. The CalDAV query allows the specification of a number of DAV properties. Specification of these
547 properties, with the exception of DAV:getetag, is considered an error in CalWS.
- 548 4. The CalDAV:propnames element is invalid

549 With those differences, the CalDAV specification is the normative reference for this operation.

550 9.1 Limiting data returned

551 This is achieved by specifying one of the following

- 552 • CalDAV:allprop return all properties (some properties are specified as not being part of the allprop
553 set so are not returned)
- 554 • CalDAV:prop An element which contains a list of properties to be returned . May only contain
555 DAV:getetag and CalDAV:calendar-data

556 Of particular interest, and complexity, is the calendar-data property which can contain a time range to limit
557 the range of recurrences returned and/or a list of calendar properties to return.

558 9.2 Pre/postconditions for calendar queries

559 The preconditions as defined in in **[RFC 4791]** Section 7.8 apply here. CalDav errors may be reported by
560 the service when preconditions or postconditions are violated.

561 9.3 Example: time range limited retrieval

562 This example shows the time-range limited retrieval from a calendar which results in 2 events, one a
563 recurring event and one a simple non-recurring event.

```
564 >> Request <<
565
566 POST /user/fred/calendar/ HTTP/1.1
567 Host: calWS.example.com
568 Depth: 1
569 Content-Type: application/xml; charset="utf-8"
570 Content-Length: xxxx
571
572 <?xml version="1.0" encoding="utf-8" ?>
573 <C:calendar-query xmlns:D="DAV:"
574                 xmlns:C="urn:ietf:params:xml:ns:caldav">
575   <D:prop>
576     <D:getetag/>
577     <C:calendar-data content-type="application/xml+calendar" >
578       <C:comp name="VCALENDAR">
```

```

579     <C:prop name="VERSION"/>
580     <C:comp name="VEVENT">
581         <C:prop name="SUMMARY"/>
582         <C:prop name="UID"/>
583         <C:prop name="DTSTART"/>
584         <C:prop name="DTEND"/>
585         <C:prop name="DURATION"/>
586         <C:prop name="RRULE"/>
587         <C:prop name="RDATE"/>
588         <C:prop name="EXRULE"/>
589         <C:prop name="EXDATE"/>
590         <C:prop name="RECURRENCE-ID"/>
591     </C:comp>
592 </C:comp>
593 </C:calendar-data>
594 </D:prop>
595 <C:filter>
596     <C:comp-filter name="VCALENDAR">
597         <C:comp-filter name="VEVENT">
598             <C:time-range start="20060104T000000Z"
599                 end="20060105T000000Z"/>
600         </C:comp-filter>
601     </C:comp-filter>
602 </C:filter>
603 </C:calendar-query>
604
605 >> Response <<
606
607 HTTP/1.1 207 Multi-Status
608 Date: Sat, 11 Nov 2006 09:32:12 GMT
609 Content-Type: application/xml; charset="utf-8"
610 Content-Length: xxxx
611
612 <?xml version="1.0" encoding="utf-8" ?>
613 <D:multistatus xmlns:D="DAV:"
614     xmlns:C="urn:ietf:params:xml:ns:caldav">
615     <D:response>
616         <D:href>http://cal.example.com/bernard/work/abcd2.ics</D:href>
617         <D:propstat>
618             <D:prop>
619                 <D:getetag>"fffff-abcd2"</D:getetag>
620                 <C:calendar-data content-type="application/xml+calendar" >
621                     <xc:icalendar
622                         xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
623     <xc:vcalendar>
624         <xc:properties>
625             <xc:calscale><text>GREGORIAN</text></xc:calscale>
626             <xc:prodid>
627                 <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
628             </xc:prodid>
629             <xc:version><xc:text>2.0</xc:text></xc:version>
630         </xc:properties>
631         <xc:components>
632             <xc:vevent>
633                 <xc:properties>
634                     <xc:dtstart>
635                         <xc:parameters>
636                             <xc:tzid>US/Eastern<xc:tzid>
637                         <xc:parameters>
638                             <xc:date-time>20060102T120000</xc:date-time>
639                     </xc:dtstart>
640                     <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
641                     <xc:summary>
642                         <xc:text>Event #2</xc:text>

```



```

643     </xc:summary>
644     <xc:uid>
645       <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
646     </xc:uid>
647     <xc:rrule>
648       <xc:recur>
649         <xc:freq>DAILY</xc:freq>
650         <xc:count>5</xc:count>
651       </xc:recur>
652     </xc:rrule>
653   </xc:properties>
654 </xc:vevent>
655
656 <xc:vevent>
657   <xc:properties>
658     <xc:dtstart>
659       <xc:parameters>
660         <xc:tzid>US/Eastern<xc:tzid>
661       <xc:parameters>
662         <xc:date-time>20060104T140000</xc:date-time>
663     </xc:dtstart>
664     <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
665     <xc:summary>
666       <xc:text>Event #2 bis</xc:text>
667     </xc:summary>
668     <xc:uid>
669       <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
670     </xc:uid>
671     <xc:recurrence-id>
672       <xc:parameters>
673         <xc:tzid>US/Eastern<xc:tzid>
674       <xc:parameters>
675         <xc:date-time>20060104T120000</xc:date-time>
676     </xc:recurrence-id>
677     <xc:rrule>
678       <xc:recur>
679         <xc:freq>DAILY</xc:freq>
680         <xc:count>5</xc:count>
681       </xc:recur>
682     </xc:rrule>
683   </xc:properties>
684 </xc:vevent>
685
686 <xc:vevent>
687   <xc:properties>
688     <xc:dtstart>
689       <xc:parameters>
690         <xc:tzid>US/Eastern<xc:tzid>
691       <xc:parameters>
692         <xc:date-time>20060106T140000</xc:date-time>
693     </xc:dtstart>
694     <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
695     <xc:summary>
696       <xc:text>Event #2 bis bis</xc:text>
697     </xc:summary>
698     <xc:uid>
699       <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
700     </xc:uid>
701     <xc:recurrence-id>
702       <xc:parameters>
703         <xc:tzid>US/Eastern<xc:tzid>
704       <xc:parameters>
705         <xc:date-time>20060106T120000</xc:date-time>
706     </xc:recurrence-id>

```

```

707     <xc:rrule>
708         <xc:recur>
709             <xc:freq>DAILY</xc:freq>
710             <xc:count>5</xc:count>
711         </xc:recur>
712     </xc:rrule>
713 </xc:properties>
714 </xc:vevent>
715 </xc:components>
716 </xc:vcalendar>
717 </xc:icalendar>
718     </C:calendar-data>
719 </D:prop>
720     <D:status>HTTP/1.1 200 OK</D:status>
721 </D:propstat>
722 </D:response>
723 <D:response>
724     <D:href>http://cal.example.com/bernard/work/abcd3.ics</D:href>
725     <D:propstat>
726         <D:prop>
727             <D:getetag>"fffff-abcd3"</D:getetag>
728             <C:calendar-data content-type="application/xml+calendar" >
729                 <xcal:icalendar
730                     xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
731 <xc:vcalendar>
732     <xc:properties>
733         <xc:calscale><text>GREGORIAN</text></xc:calscale>
734         <xc:prodid>
735             <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
736         </xc:prodid>
737         <xc:version><xc:text>2.0</xc:text></xc:version>
738     </xc:properties>
739     <xc:components>
740     <xc:vevent>
741     <xc:properties>
742     <xc:dtstart>
743         <xc:parameters>
744             <xc:tzid>US/Eastern<xc:tzid>
745         <xc:parameters>
746             <xc:date-time>20060104T100000</xc:date-time>
747         </xc:dtstart>
748     <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
749     <xc:summary>
750         <xc:text>Event #3</xc:text>
751     </xc:summary>
752     <xc:uid>
753         <xc:text>DC6C50A017428C5216A2F1CD@example.com</xc:text>
754     </xc:uid>
755     <xc:rrule>
756         <xc:recur>
757             <xc:freq>DAILY</xc:freq>
758             <xc:count>5</xc:count>
759         </xc:recur>
760     </xc:rrule>
761     </xc:properties>
762 </xc:vevent>
763 </xc:components>
764 </xc:vcalendar>
765 </xc:icalendar>
766     </C:calendar-data>
767 </D:prop>
768     <D:status>HTTP/1.1 200 OK</D:status>
769 </D:propstat>
770 </D:response>

```

772 10 Free-busy queries

773 Free-busy queries are used to obtain free-busy information for a calendar-collection or principals. The
774 result contains information only for events to which the current principal has sufficient access.

775 When targeted at a calendar collection the result is based only on the calendaring entities contained in
776 that collection. When targeted at a principal free-busy URL the result will be based on all information
777 which affect the principals free-busy status, for example availability.

778 The possible targets are:

- 779 • A calendar collection URL
- 780 • The XRD link with relation CalWS/current-principal-freebusy
- 781 • The XRD link with relation CalWS/principal-freebusy with a principal given in the request.

782 The query follows the specification defined in **[FreeBusy Read URL]** with certain limitations. As an
783 authenticated user to the CalWS service scheduling read-freebusy privileges must have been granted. As
784 an unauthenticated user equivalent access must have been granted to unauthenticated access.

785 Freebusy information is returned by default as xcalendar vfreebusy components, as defined by **[draft-
786 xcal]**. Such a component is not meant to conform to the requirements of VFREEBUSY components in
787 **[RFC 5546]**. The VFREEBUSY component SHOULD conform to section "4.6.4 Free/Busy Component" of
788 **[RFC 5545]**. A client SHOULD ignore the ORGANIZER field..

789 Since a Freebusy query can only refer to a single user, a client will already know how to match the result
790 component to a user. A server MUST only return a single vfreebusy component.

791 10.1 ACCEPT header

792 The Accept header is used to specify the format for the returned data. In the absence of a header the
793 data should be returned as specified in **[draft-xcal]**, that is, as if the following had been specified

794 `ACCEPT: application/xml+calendar`

795 10.2 URL Query Parameters

796 None of these parameters are required except for the conditions noted below. Appropriate defaults will be
797 supplied by the server.

798 10.2.1 start

799 **Default:** The default value is left up to the server. It may be the current day, start of the current
800 month, etc.

801 **Description:** Specifies the start date for the Freebusy data. The server is free to ignore this value and
802 return data in any time range. The client must check the data for the returned time range.

803 **Format:** A profile of an **[RFC3339]** Date/Time. Fractional time is not supported. The server MUST
804 support the expanded version e.g.

805 `2007-01-02T13:00:00-08:00`

806 It is up to the server to interpret local date/times.

807 **Example:**

808 `2007-02-03T15:30:00-0800`

809 `2007-12-01T10:15:00Z`

810 **Notes:** Specifying only a start date/time without specifying an end-date/time or period should be
811 interpreted as in **[RFC 5545]**. The effective period should cover the remainder of that day.

812 Date-only values are disallowed as the server cannot determine the correct start of the day. Only
813 UTC or date/time with offset values are permitted.

814 10.2.2 end

815 **Default:** Same as start

816 **Description:** Specifies the end date for the Freebusy data. The server is free to ignore this value.

817 **Format:** Same as start

818 **Example:** Same as start

819 10.2.3 period

820 **Default:** The default value is left up to the server. The recommended value is "P42D".

821 **Description:** Specifies the amount of Freebusy data to return. A client cannot specify both a period
822 and an end date. Period is relative to the start parameter.

823 **Format:** A duration as defined in section 4.3.6 of [RFC 5545]

824 **Example:**

825 `P42D`

826 10.2.4 account

827 **Default:** none

828 **Description:** Specifies the principal when the request is targeted at the XRD CalWS/principal-
829 freebusy. Specification of this parameter is an error otherwise.

830 **Format:** Server specific

831 **Example:**

832 `fred`
833 `/principals/users/jim`
834 `user1@example.com`

835 10.3 URL parameters - notes

836 The server is free to ignore the start, end and period parameters. It is recommended that the server return
837 at least 6 weeks of data from the current day.

838 A client MUST check the time range in the VFREEBUSY response as a server may return a different time
839 range than the requested range.

840 10.4 HTTP Operations

841 The server SHOULD return an Etag response header for a successful GET request targeting a Freebusy
842 read URL. Clients MAY use the Etag response header value to do subsequent "conditional" GET
843 requests that will avoid re-sending the Freebusy data again if it has not changed.

844 10.5 Response Codes

845 Below are the typical status codes returned by a GET request targeting a Free-busy URL. Note that other
846 HTTP status codes not listed here might also be returned by a server.

- 847 • 200 OK
- 848 • 302 Found
- 849 • 400 Start parameter could not be understood / End parameter could not be understood / Period
850 parameter could not be understood
- 851 • 401 Unauthorized
- 852 • 403 Forbidden
- 853 • 404 The data for the requested principal is not currently available, but may be available later.
- 854 • 406 The requested format in the accept header is not supported.
- 855 • 410 The data for the requested principal is no longer available

856 • 500 General server error

857 10.6 Examples

858 The following are examples of URLs used to retrieve Free-busy data for a user:

```
859 http://www.example.com/freebusy/user1@example.com?
860 start=2007-09-01T00:00:00-08:00&end=2007-09-31T00:00:00-08:00
861
862 http://www.example.com/freebusy/user1@example.com?
863 start=2007-09-01T00:00:00-08:00&end=2007-09-31T00:00:00-08:00
864
865 http://www.example.com/freebusy/user1@example.com
866
867 http://www.example.com/freebusy?user=user%201@example.com&
868 start=2008-01-01T00:00:00Z&end=2008-12-31T00:00:00Z
```

869 Some Request/Response Examples:

870 A URL with no query parameters:

```
871 >> Request <<
872 GET /freebusy/bernard/ HTTP/1.1
873 Host: www.example.com
874
875 >> Response <<
876 HTTP/1.1 200 OK
877 Content-Type: application/xml+calendar; charset="utf-8"
878 Content-Length: xxxx
879
880 <xc:icalendar xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
881   <xc:vcalendar>
882     <xc:properties>
883       <xc:calscale><text>GREGORIAN</text></xc:calscale>
884       <xc:prodid>
885         <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
886       </xc:prodid>
887       <xc:version><xc:text>2.0</xc:text></xc:version>
888     </xc:properties>
889     <xc:components>
890       <xc:vfreebusy>
891         <xc:properties>
892           <xc:uid>
893             <xc:text>76ef34-54a3d2@example.com</xc:text>
894           </xc:uid>
895           <xc:dtstart>
896             <xc:date-time>20060101T000000Z</xc:date-time>
897           </xc:dtstart>
898           <xc:dtend>
899             <xc:date-time>20060108T000000Z</xc:date-time>
900           </xc:dtend>
901           <xc:dtstamp>
902             <xc:date-time>20050530T123421Z</xc:date-time>
903           </xc:dtstamp>
904           <xc:freebusy>
905             <xc:parameters>
906               <xc:fbsytype>BUSYTENTATIVE</xc:fbsytype>
907             </xc:parameters>
908             <xc:period>20060102T100000Z/20060102T120000Z</xc:period>
909           </xc:freebusy>
910           <xc:freebusy>
911             <xc:period>20060103T100000Z/20060103T120000Z</xc:period>
912           </xc:freebusy>
913           <xc:freebusy>
914             <xc:period>20060104T100000Z/20060104T120000Z</xc:period>
915           </xc:freebusy>
```

```

916         <xc:freebusy>
917             <xc:parameters>
918                 <xc:fbtype>BUSYUNAVAILABLE<xc:fbtype>
919             <xc:parameters>
920             <xc:period>20060105T100000Z/20060105T120000Z</xc:period>
921         </xc:freebusy>
922     <xc:freebusy>
923         <xc:period>20060106T100000Z/20060106T120000Z</xc:period>
924     </xc:freebusy>
925 </xc:vfreebusy>
926 </xc:components>
927 </xc:vcalendar>
928 <xc:icalendar>

```

929 A URL with start and end parameters:

```

930 >> Request <<
931 GET /freebusy/user1@example.com?start=2007-09-01T00:00:00-08:00&end=2007-09-
932 31T00:00:00-08:00
933 HTTP/1.1
934 Host: www.example.com
935
936 >> Response <<
937 HTTP/1.1 200 OK
938 Content-Type: application/xml+calendar; charset="utf-8"
939 Content-Length: xxxx
940
941 <xc:icalendar xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
942     <xc:vcalendar>
943         <xc:properties>
944             <xc:calscale><text>GREGORIAN</text></xc:calscale>
945             <xc:prodid>
946                 <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
947             </xc:prodid>
948             <xc:version><xc:text>2.0</xc:text></xc:version>
949         </xc:properties>
950         <xc:components>
951             <xc:vfreebusy>
952                 <xc:properties>
953                     <xc:uid>
954                         <xc:text>76ef34-54a3d2@example.com</xc:text>
955                     </xc:uid>
956                     <xc:dtstart>
957                         <xc:date-time>20070901T000000Z</xc:date-time>
958                     </xc:dtstart>
959                     <xc:dtend>
960                         <xc:date-time>20070931T000000Z</xc:date-time>
961                     </xc:dtend>
962                     <xc:dtstamp>
963                         <xc:date-time>20050530T123421Z</xc:date-time>
964                     </xc:dtstamp>
965                     <xc:freebusy>
966                         <xc:period>20070915T230000Z/20070916T010000Z</xc:period>
967                     </xc:freebusy>
968                 </xc:vfreebusy>
969             </xc:components>
970         </xc:vcalendar>
971     <xc:icalendar>

```

972 A URL for which the server does not have any data for that user:

```

973 >> Request <<
974 GET /freebusy/user1@example.com?start=2012-12-01T00:00:00-08:00&end=2012-12-
975 31T00:00:00-08:00
976 HTTP/1.1
977 Host: www.example.com
978

```

979
980
981

```
>> Response <<  
HTTP/1.1 404 No data
```

982 11 Conformance

983 Certain calendaring properties and components are interrelated and it is necessary to have knowledge of
984 all these properties and their current values to allow consistent update and understanding of a target
985 component. The normative definition for these relationships is RFC5445, RFC5446 and related RFCs.

986 As in those specifications this REST-ful protocol assumes a complete view of entities being fetched or
987 updated. This is necessary to ensure that properties are not lost when round tripped through a service or
988 client. To this end all parties in any RESTful transaction MUST preserve any data they do not understand.
989 This allows the data model to be updated by the addition of properties, parameters and value types.

990 Services allowing updates to entities MUST ensure that the result after an update operation is still
991 internally consistent.

992 11.1 Start, end and duration in calendar components

993 A period of time is fully specified by a start and an end or duration.

994 11.1.1 Updating, transporting and maintaining start, and and duration.

- 995 • For all components the calculated or specified start must be at or before the end.
- 996 • When a system updates or stores a calendar component it MUST retain the relationship of start,
997 end and duration. Applications MUST NOT without good cause, change a start and end pair into
998 a start and duration nor the reverse. Semantically they are not equivalent when DST transitions
999 occur during the time of the event.
- 1000 • For interoperability, iCalendar based systems SHOULD avoid the use of weekly durations and
1001 XML based systems SHOULD avoid the use of yearly durations.

1002 11.1.2 VEVENT:

- 1003 • The three properties are DTSTART, DTEND and DURATION.
- 1004 • DTSTART MUST appear once and only one of DTEND or DURATION MAY be present.
- 1005 • The DTSTART property for a VEVENT specifies the inclusive start of the event. For recurring
1006 events, it also specifies the very first instance in the recurrence set.
- 1007 • The DTEND property for a VEVENT calendar component specifies the non-inclusive end of the
1008 event.
- 1009 • For cases where a VEVENT calendar component specifies a DTSTART property with a DATE
1010 value type but no DTEND nor DURATION property, the event's duration is taken to be one day.
- 1011 • For cases where a VEVENT calendar component specifies a DTSTART property with a DATE-
1012 TIME value type but no DTEND nor DURATION property, the event ends on the same calendar
1013 date and time of day specified by the DTSTART property, that is, it signifies a zero length instant
1014 in time.

1015 11.1.3 VTODO:

- 1016 • The three properties are DTSTART, DUE, DURATION.
- 1017 • DTSTART MAY appear once.
- 1018 • Either DUE or DURATION MAY appear in a VTODO, but DUE and DURATION MUST NOT
1019 occur in the same VTODO.
- 1020 • If DURATION does appear in a VTODO, then DTSTART MUST also appear in the same VTODO.

- 1021 • The three properties for a VTODD are related in the same way as for VEVENT. Additionally a
1022 VTODD calendar component without the DTSTART and DUE (or DURATION) properties
1023 specifies a VTODD that will be associated with each successive calendar date, until it is
1024 completed.

1025 **11.1.4 VJOURNAL:**

- 1026 • DTSTART only, which may be a date or date-time value.

1027 **11.1.5 VAVAILABILITY**

- 1028 • DTSTART and DTEND if specified MUST be date-time values.
1029 • DTSTART MAY appear once and signifies start of the busy period.
1030 • Only one of DTEND or DURATION MAY appear and signify the end of the busy period.
1031 • If DURATION does appear in a VAVAILABILITY, then DTSTART MUST also appear in the same
1032 VAVAILABILITY.

1033 **11.1.6 AVAILABILITY**

- 1034 ● DTSTART and DTEND if specified MUST be date-time values.
1035 ● DTSTART MUST appear once and signifies start of the free period.
1036 ● Only one of DTEND or DURATION MAY appear and signify the end of the free period.

1037 **11.2 Recurrences.**

- 1038 • The RECURRENCE-ID is a property of each instance of a recurring event. It is calculated from
1039 the DTSTART and the recurrence rules or added to the set by the RDATE property.
1040 • RDATE, EXDATE and RECURRENCE-ID must take the same form as the DTSTART. That is if
1041 DTSTART is a DATE value then the RDATE and EXDATE must be DATE. If DTSTART is a date-
1042 time the RDATE and EXDATE values must take the same form, including the same timezone.
1043 • Overrides to an instance are specified by completely specifying the instance with the appropriate
1044 RECURRENCE-ID property.
1045 • An RDATE adds an instance to the recurrence set.
1046 • An EXDATE deletes an instance by specifying the recurrence id(s) to be deleted. Applications
1047 SHOULD NOT specify overrides for instances so deleted.
1048 • The recurrence set is calculated from the RRULE and RDATES and then applying any EXDATE
1049 properties. That is EXDATE takes precedence over RDATE and the RRULE.

1050 **11.3 Alarms:**

- 1051 • Alarms are typically anchored to the start or end of an event or task. This is defined by the
1052 RELATED parameter to the TRIGGER property.

1053 **11.4 Unrecognized or unsupported elements**

- 1054 • A system SHOULD reject any attempt to store components which it does not support. A SYSTEM
1055 MUST respond with a CalWS:unsupported-calendar-component if such an attempt is made.
1056 • A system MUST ignore but preserve any elements it does not understand.
1057

1058 **Appendix A. Acknowledgments**

1059 The following individuals have participated in the creation of this specification and are gratefully
1060 acknowledged:

1061 **Participants:**

- 1062 Bruce Bartell, Southern California Edison
- 1063 Brad Benson, Trane
- 1064 Edward Cazalet, Individual
- 1065 Toby Considine, University of North Carolina at Chapel Hill
- 1066 William Cox, Individual
- 1067 Sharon Dinges, Trane
- 1068 Mike, Douglass, Rensselaer Polytechnic Institute
- 1069 Craig Gemmill, Tridium, Inc.
- 1070 Girish Ghatikar, Lawrence Berkeley National Laboratory
- 1071 Gerald Gray, Southern California Edison
- 1072 David Hardin, ENERNOC
- 1073 Gale Horst, Electric Power Research Institute (EPRI)
- 1074 Gershon Janssen, Individual
- 1075 Ed Koch, Akuacom Inc.
- 1076 Benoit Lepeuple, LonMark International*
- 1077 Carl Mattocks, CheckMi*
- 1078 Robert Old, Siemens AG
- 1079 Alexander Papaspyrou, Technische Universitat Dortmund
- 1080 Joshua Phillips, ISO/RTO Council (IRC)
- 1081 Jeremy J. Roberts, LonMark International
- 1082 David Thewlis, CalConnect

1083
1084 The Calendaring and Scheduling Consortium (CalConnect) TC-XML committee worked closely with WS-
1085 Calendar Technical Committee, bridging to developing IETF standards and contributing the services
1086 definitions that make up Services in Section 4. The Technical Committee gratefully acknowledges their
1087 assistance and cooperation as well. Contributors to TC XML include:

- 1088 Cyrus Daboo, Apple
 - 1089 Mike Douglass, Rensselaer Polytechnic Institute
 - 1090 Steven Lees, Microsoft
 - 1091 Tong Li, IBM
- 1092

1093

Appendix B. An Introduction to Internet Calendaring

1094 *The WS-Calendar Technical Committee thanks CalConnect for contributing this overview of iCalendar*
1095 *and its use.*

1096 B.1 icalendar

1097 B.1.1 History

1098 The iCalendar specification was first produced by the IETF in 1998 as RFC 2445 [1]. Since then it has
1099 become the dominant standard for calendar data interchange on the internet and between devices
1100 (desktop computers, mobile phones etc.). The specification was revised in 2009 as RFC 5545 [4].

1101 Alongside iCalendar is the iTIP specification (RFC 2446 [2] and revised as RFC 5546[5]) that defines how
1102 iCalendar is used to carry out scheduling operations (for example, how an organizer can invite attendees
1103 to a meeting and receive their replies). This forms the basis for email-based scheduling using iMIP (the
1104 specification that describes how to use iTIP with email - RFC 6047 [3]).

1105 iCalendar itself is a text-based data format. However, an XML format is also available, providing a one-to-
1106 one mapping to the text format (draft [7]).

1107 iCalendar data files typically have a .ics file name extension. Most desktop calendar clients can import or
1108 export iCalendar data, or directly access such data over the Internet using a variety of protocols.

1109 B.1.2 Data model

1110 The iCalendar data format has a well defined data model. "iCalendar objects" encompass a set of
1111 "iCalendar Components" each of which contains a set of "iCalendar properties" and possibly other sub-
1112 Components. An iCalendar property consists of a name, a set of optional parameters (specified as "key-
1113 value" pairs) and a value.

1114 iCalendar Components include:

1115 "VEVENT" which represents an event

1116 "VTODO" which represents a task or to-do

1117 "VJOURNAL" which represents a journal entry

1118 "VFREEBUSY" which represents periods of free or busy time information

1119 "VTIMEZONE" which represents a timezone definition (timezone offset and daylight saving rules)

1120 "VALARM" is currently the only defined sub-Component and is used to set alarms or reminders on events
1121 or tasks.

1122 Properties include:

1123 "DTSTART" which represents a start time for a Component

1124 "DTEND" which represents an end time for a Component

1125 "SUMMARY" which represents a title or summary for a Component

1126 "RRULE" which can specify rules for repeating events or tasks (for example, every day, every week on
1127 Tuesdays, etc.)

1128 "ORGANIZER" which represents the calendar user who is organizing an event or assigning a task

1129 "ATTENDEE" which represents calendar users attending an event or assigned a task

1130 In addition to this data model and the pre-defined properties, the specification defines how all those are
1131 used together to define the semantics of calendar objects and scheduling. The semantics are basically a
1132 set of rules stating how all the Components and properties are used together to ensure that all iCalendar
1133 products can work together to achieve good interoperability. For example, a rule requires that all events
1134 must have one and only one "DTSTART" property. The most important part of the iCalendar specification

1135 is the semantics of the calendaring model that it represents. The use of text or XML to encode those is
1136 secondary.

1137 **B.1.3 Scheduling**

1138 The iTIP specification defines how iCalendar objects are exchanged in order to accomplish the key task
1139 needed to schedule events or tasks. An example of a simple workflow is as follows:

- 1140 1. To schedule an event, an organizer creates the iCalendar object representing the event and adds
1141 calendar users as attendees.
- 1142 2. The organizer then sends an iTIP "REQUEST" message to all the attendees.
- 1143 3. Upon receipt of the scheduling message, each attendee can decide whether they want to attend
1144 the meeting or not.
- 1145 4. Each attendee can then respond back to the organizer using an iTIP "REPLY" message
1146 indicating their own attendance status.

1147 iTIP supports other types of scheduling messages, for example, to cancel meetings, add new instances to
1148 a repeating meeting, etc.

1149 **B.1.4 Extensibility**

1150 iCalendar was designed to be extensible, allowing for new Components, properties and parameters to be
1151 defined as needed. A registry exists to maintain the list of standard extensions with references to their
1152 definitions to ensure anyone can use them and work well with others.

1153 **B.2 Calendar data access and exchange protocols**

1154 **B.2.1 Internet Calendar Subscriptions**

1155 An Internet calendar subscription is simply an iCalendar data file made available on a web server. Users
1156 can use this data in two ways:

- 1157 – The data can be downloaded from the web server and then imported directly into an iCalendar
1158 aware client. This solution works well for calendar data that is not likely to change over time (for
1159 example the list of national holidays for the next year).
- 1160 – Calendar clients that support "direct" subscriptions can use the URL to the calendar data on the
1161 web server to download the calendar data themselves. Additionally, the clients can check the web
1162 server on a regular basis for updates to the calendar data, and then update their own cached
1163 copy of it. This allows calendar data that changes over time to be kept synchronized.

1164 **B.2.2 CalDAV**

1165 CalDAV is a calendar access protocol and is defined in RFC 4791 [6]. The protocol is based on WebDAV
1166 which is an extension to HTTP that provides enhanced capabilities for document management on web
1167 servers.

1168 CalDAV is used in a variety of different environments, ranging from very large internet service providers,
1169 to large and small corporations or institutions, and to small businesses and individuals.

1170 CalDAV clients include desktop applications, mobile devices and browser-based solutions. It can also be
1171 used by "applets", for example, a web page panel that displays a user's upcoming events.

1172 One of the key aspects of CalDAV is its data model. Simply put, it defines a "calendar home" for each
1173 calendar user, within which any number of "calendars" can be created. Each "calendar" can contain any
1174 number of iCalendar objects representing individual events, tasks or journal entries. This data model
1175 ensures that clients and servers can interoperate well.

1176 In addition to providing simple operations to read, write and delete calendar data, CalDAV provides a
1177 querying mechanism to allow clients to fetch calendar data matching specific criteria. This is commonly

1178 used by clients to do "time-range" queries, i.e., find the set of events that occur within a given start/end
1179 time period.

1180 CalDAV also supports access control allowing for features such as delegated calendars and calendar
1181 sharing.

1182 CalDAV also specifies how scheduling operations can be done using the protocol. Whilst it uses the
1183 semantics of the iTIP protocol, it simplifies the process by allowing simple calendar data write operations
1184 to trigger the sending of scheduling messages, and it has the server automatically process the receipt of
1185 scheduling messages. Scheduling can be done with other users on the CalDAV server or with calendar
1186 users on other systems (via some form of "gateway").

1187 **B.2.3 ActiveSync/SyncML**

1188 ActiveSync and SyncML are technologies that allow multiple devices to synchronize data with a server,
1189 with calendar data being one of the classes of data supported. These have typically been used for low-
1190 end and high-end mobile devices.

1191 **B.2.4 CalWS**

1192 CalWS refers to a set of web services calendar access APIs developed under a cooperative agreement
1193 between The Calendaring and Scheduling Consortium (CalConnect) and OASIS, and being published as
1194 a work product of the WS-Calendar Technical Committee. CalWS defines an API to access and
1195 manipulate calendar data stored on a server. It follows a similar data model to CalDAV and has been
1196 designed to co-exist with a CalDAV service offering the same data.

1197 This specification is part of the CalWS set.

1198 **B.2.5 iSchedule**

1199 iSchedule is a protocol to allow scheduling between users on different calendaring systems and across
1200 different internet domains. It transports iTIP scheduling messages using HTTP between servers. Servers
1201 use DNS and various security mechanisms to determine the authenticity of messages received.

1202 It has been specifically designed to be independent of any calendar system in use at the endpoints, so
1203 that it is compatible with many different systems. This allows organizations with different calendar
1204 systems to exchange scheduling messages with each other, and also allows a single organization with
1205 multiple calendar systems (for example due to mergers, or different departmental requirements) to
1206 exchange scheduling messages between users of each system.

1207 **B.3 References**

1208 [1] <https://datatracker.ietf.org/doc/rfc2445/> : 'Internet Calendaring and Scheduling Core Object
1209 Specification'

1210 [2] <https://datatracker.ietf.org/doc/rfc2446/> : 'iCalendar Transport-Independent Interoperability Protocol'

1211 [3] <https://datatracker.ietf.org/doc/rfc6047/> : 'iCalendar Message-Based Interoperability Protocol'

1212 [4] <https://datatracker.ietf.org/doc/rfc5545/> : 'Internet Calendaring and Scheduling Core Object
1213 Specification'

1214 [5] <https://datatracker.ietf.org/doc/rfc5546/> : 'iCalendar Transport-Independent Interoperability Protocol'

1215 [6] <https://datatracker.ietf.org/doc/rfc4791/> : 'Calendaring Extensions to WebDAV'

1216 [7] <https://datatracker.ietf.org/doc/draft-daboo-et-al-icalendar-in-xml/> : 'xCal: The XML format for
1217 iCalendar'

1218

1219

Appendix C. Revision History

Revision	Date	Editor	Changes Made
ws-calendar-wd19	19-Mar-2011	Toby Considine	Originally contributed by Mike Douglass as part of WS-Calendar v1.0 Specification. See full history in that document.
WD02	13-Feb-2012	Toby Considine	Ported to separate document. "Promoted" all section headers.
WD03	15 Feb-2012	Toby Considine	Added Intro, updated namespaces to meet OASIS standard
WD04	17 February 2012	Toby Considine	Additional namespace clean-up in response to Cover comments. Consistent capitalization of calWS when used as a namespace identifier Clean-up of CalWS discussion in appendix
WD05	17 February 2012	Toby Considine	Types, capitalization, missing XRD reference
WD06	29 January 2013	Mike Douglas	Added Conformance