JMAP for Tasks

Abstract

This document specifies a data model for synchronizing task data
with a server using JMAP.

Status of This Memo

This Internet-Draft is submitted in full conformance with the
provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering
Task Force (IETF). Note that other groups may also distribute
working documents as Internet-Drafts. The list of current Internet-
Drafts is at https://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 2 August 2021.

Copyright Notice

Table of Contents

1. Introduction

JMAP ([RFC8620] - JSON Meta Application Protocol) is a generic protocol for synchronizing data, such as mail, calendars or contacts, between a client and a server. It is optimized for mobile and web environments, and aims to provide a consistent interface to different data types.

JMAP for Calendars ([I-D.ietf-jmap-calendars]) defines a data model for synchronizing calendar data between a client and a server using JMAP. The data model is designed to allow a server to provide consistent access to the same data via CalDAV [RFC4791] as well as JMAP.

While CalDAV defines access to tasks, JMAP for Calendars does not. This specification fills this gap and defines a data model for synchronizing task data between a client and a server using JMAP. It is built upon JMAP for Calendars and reuses most of its definitions. For better readability this document only outlines differences between this specification and JMAP for Calendars. If not stated otherwise, the same specifics that apply to Calendar, CalendarEvent and CalendarEventNotification objects as defined in the aforemetioned specification also apply to similar data types introduced in this specification.

1.1. Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Type signatures, examples, and property descriptions in this document follow the conventions established in Section 1.1 of [RFC8620]. Data types defined in the core specification are also used in this document.

1.2. Terminology

The same terminology is used in this document as in the core JMAP specification, see [RFC8620], Section 1.6.

The terms ParticipantIdentity, TaskList, Task and TaskNotification are used to refer to the data types defined in this document and instances of those data types.

1.3. Data Model Overview

Similar to JMAP for Calendar, an Account (see [RFC8620], Section 1.6.2) contains zero or more TaskList objects, which is a named collection of Tasks belonging to a Principal (see [I-D.jenkins-jmap-sharing] Section XXX). Task lists can also provide defaults, such as alerts and a color to apply to tasks in the calendar. Clients commonly let users toggle visibility of tasks belonging to a particular task list on/off. Servers may allow a task to belong to multiple TaskLists within an account.

A Task is a representation of a single task or recurring series of Tasks in JSTask [I-D.ietf-calext-jscalendar] format. Recurrence rules and alerts as defined in JMAP for Calendars (see [I-D.ietf-jmap-calendars] Section XXX) apply.

Just like the CalendarEventNotification objects (see [I-D.ietf-jmap-calendars] Section XXX), TaskNotification objects keep track of the history of changes made to a task by other users. Similarly, the ShareNotification type (see [I-D.jenkins-jmap-sharing] Section XXX) notifies the user when their access to another user's calendar is granted or revoked.

1.4. Addition to the Capabilities Object

The capabilities object is returned as part of the JMAP Session object; see [RFC8620], Section 2. This document defines one additional capability URI.

1.4.1. urn:ietf:params:jmap:tasks

This represents support for the TaskList, Task and TaskNotification data types and associated API methods. The value of this property in the JMAP Session capabilities property is an empty object.

The value of this property in an account's accountCapabilities property is an object that MUST contain the following information on server capabilities and permissions for that account:

* **shareesActAs**: String This MUST be one of:

  - self - sharees act as themselves when using tasks in this account.

  - secretary- sharees act as the principal to which this account belongs.

* **maxTaskListsPerTask**: UnsignedInt|null The maximum number of TaskLists (see Section XXX) that can be can assigned to a single Task object (see Section XXX). This MUST be an integer >= 1, or

null for no limit (or rather, the limit is always the number of TaskLists in the account).

* **minDateTime**: LocalDate The earliest date-time the server is willing to accept for any date stored in a Task.

* **maxDateTime**: LocalDate The latest date-time the server is willing to accept for any date stored in a Task.

* **maxExpandedQueryDuration**: Duration The maximum duration the user may query over when asking the server to expand recurrences.

* **maxAssigneesPerTask**: Number|null The maximum number of assignees a single task may have, or null for no limit.

* **mayCreateTaskList**: Boolean If true, the user may create a task list in this account.

2. Principals

For systems that also support JMAP Sharing [RFC XXX], the tasks capability is used to indicate that this principal may be used with tasks.

2.1. Principal Capability urn:ietf:params:jmap:tasks

A "urn:ietf:params:jmap:tasks" property is added to the Principal "capabilities" object, the value of which is an object with the following properties:

* **accountId**: Id|null Id of Account with the urn:ietf:params:jmap:tasks capability that contains the task data for this principal, or null if none (e.g. the Principal is a group just used for permissions management), or the user does not have access to any data in the account.

* **account**: Account|null The JMAP Account object corresponding to the accountId, null if none.

* **sendTo**: String[String]|null If this principal may be added as a participant to an event, this is the map of methods for adding it, in the same format as Participant#sendTo in JSTask (see [I-D.ietf-calext-jscalendar], Section 4.4.5).

3. Assignee Identities

An AssigneeIdentity stores information about a URI that represents
the user within that account in an event's assignees. It has the
following properties:

   *__id__: Id (immutable; server-set) The id of the AssigneeIdentity.

   *__name__: String (default: "") The display name of the assignee to
    use when adding this assignee to a task, e.g. "Jane Bloggs".

   *__sendTo__: String[String] Represents methods by which the
    participant may receive invitations and updates to an event.

    The keys in the property value are the available methods and MUST
    only contain ASCII alphanumeric characters (A-Za-z0-9). The value
    is a URI for the method specified in the key.

An assignee in an task corresponds to an AssigneeIdentity if any of
the method/uri pairs in the sendTo property of the participant are
identical to a method/uri pair in the sendTo property of the
identity.

The following JMAP methods are supported.

3.1. AssigneeIdentity/get

This is a standard "/get" method as described in [RFC8620], Section
5.1. The *ids* argument may be null to fetch all at once.

3.2. AssigneeIdentity/changes

This is a standard "/changes" method as described in [RFC8620],
Section 5.2.

3.3. AssigneeIdentity/set

This is a standard "/set" method as described in [RFC8620], Section
5.3. The server MAY restrict the uri values the user may claim, for
example only allowing mailto: URIs with email addresses that belong
to the user. A standard forbidden error is returned to reject non-
permissible changes.

4. TaskLists

A TaskList is a named collection of tasks. All tasks are associated
with at least one TaskList.

A **TaskList** object has the following properties:

  * **id**: Id (immutable; server-set) The id of the task list.

  * **role**: String|null (default: null) Denotes the task list has a special purpose. This MUST be one of the following:

     - inbox: This is the principal's default task list;

     - trash: This task list holds messages the user has discarded;

  * **name**: String The user-visible name of the task list. This may be any UTF-8 string of at least 1 character in length and maximum 255 octets in size.

  * **description**: String|null (default: null) An optional longer-form description of the task list, to provide context in shared environments where users need more than just the name.

  * **color**: String|null (default: null) A color to be used when displaying events associated with the task list.

     If not null, the value MUST be a case-insensitive color name taken from the set of names defined in Section 4.3 of CSS Color Module Level 3 [COLORS](#), or an RGB value in hexadecimal notation, as defined in Section 4.2.1 of CSS Color Module Level 3.

     The color SHOULD have sufficient contrast to be used as text on a white background.

  * **sortOrder**: UnsignedInt (default: 0) Defines the sort order of task lists when presented in the client's UI, so it is consistent between devices. The number MUST be an integer in the range $0 <= sortOrder < 2^{31}$.

     A task list with a lower order should be displayed before a list with a higher order in any list of task lists in the client's UI. Task lists with equal order SHOULD be sorted in alphabetical order by name. The sorting should take into account locale-specific character order convention.

  * **isSubscribed**: Boolean Has the user indicated they wish to see this task list in their client? This SHOULD default to false for task lists in shared accounts the user has access to and true for any new task list created by the user themself.

     If false, the task list should only be displayed when the user explicitly requests it or to offer it for the user to subscribe to.

*   *__isVisible__: Boolean (default: true) Should the task list's events
    be displayed to the user at the moment? Clients MUST ignore this
    property if isSubscribed is false. If an event is in multiple
    task lists, it should be displayed if isVisible is true for any
    of those task lists.

*   *__defaultAlertsWithTime__: Id[Alert]|null (default: null) A map of
    alert ids to Alert objects (see [I-D.ietf-calext-jscalendar],
    Section 4.5.2) to apply for events where "showWithoutTime" is
    false and "useDefaultAlerts" is true. Ids MUST be unique across
    all default alerts in the account, including those in other task
    lists; a UUID is recommended.

*   *__defaultAlertsWithoutTime__: Id[Alert]|null (default: null) A map of
    alert ids to Alert objects (see [I-D.ietf-calext-jscalendar],
    Section 4.5.2) to apply for events where "showWithoutTime" is
    true and "useDefaultAlerts" is true. Ids MUST be unique across
    all default alerts in the account, including those in other task
    lists; a UUID is recommended.

*   *__timeZone__: String|null (default: null) The time zone to use for
    events without a time zone when the server needs to resolve them
    into absolute time, e.g., for alerts or availability calculation.
    The value MUST be a time zone id from the IANA Time Zone Database
    TZDB. If null, the timeZone of the account's associated Principal
    will be used. Clients SHOULD use this as the default for new
    events in this task list if set.

*   *__shareWith__: Id[CalendarRights]|null (default: null) A map of
    Principal id to rights for principals this calendar is shared
    with. The principal to which this task list belongs MUST NOT be
    in this set. This is null if the user requesting the object does
    not have the mayAdmin right, or if the task list is not shared
    with anyone. May be modified only if the user has the mayAdmin
    right. The account id for the principals may be found in the
    urn:ietf:params:jmap:principals:owner capability of the Account
    to which the calendar belongs.

The user is an __owner__ for a task if the Task object has an "assignee"
property, and one of the Participant objects both:

1. Has the "chair" role.

2. Corresponds to one of the user's AssigneeIdentity objects in
    the account.

A task has no owner if its assignee property is null or omitted.

TODO currently disregarding myRights

4.1. TaskList/get

This is a standard "/get" method as described in [RFC8620], Section
5.1. The *ids* argument may be null to fetch all at once.

TODO add part about rights properties.

4.2. TaskList/changes

This is a standard "/changes" method as described in [RFC8620],
Section 5.2.

4.3. TaskList/set

This is the "Calendar/set" method as described in [I-D.ietf-jmap-
calendars], Section XXX.

TODO copy+paste from "Calendar/set" and replace
onDestroyRemoveEvents by onDestroyRemoveTasks (and
calendarHasEvent).

5. Tasks

A **Task** object contains information about a task, or recurring series
of tasks. It is a JSTask object, as defined in [I-D.ietf-calext-
jscalendar], with the following additional properties:

  *   **id**: Id The id of the Task. This property is immutable. The id
       uniquely identifies a JSTask with a particular "uid" and
       "recurrenceId" within a particular account.

  *   **taskListIds**: Id[Boolean] The set of TaskList ids this task
       belongs to. An task MUST belong to one or more TaskLists at all
       times (until it is destroyed). The set is represented as an
       object, with each key being a *TaskList id*. The value for each key
       in the object MUST be true.

  *   **isDraft**: Boolean If true, this task is to be considered a draft.
       The server will not send any push notifications for alerts. This
       may only be set to true upon creation. Once set to false, the
       value cannot be updated to true. This property MUST NOT appear in
       "recurrenceOverrides".

  *   **utcStart**: UTCDate For simple clients that do not or cannot
       implement time zone support. Clients should only use this if also
       asking the server to expand recurrences, as you cannot accurately
       expand a recurrence without the original time zone.

       This property is calculated at fetch time by the server. Time
       zones are political and they can and do change at any time.

Fetching exactly the same property again may return a different
results if the time zone data has been updated on the server.
Time zone data changes are not considered "updates" to the task.

If set, server will convert to the task's current time zone using
its current time zone data and store the local time.

This is not included by default and must be requested explicitly.

Floating tasks (tasks without a time zone) will be interpreted as
per the time zone given as a Task/get argument.

Note that it is not possible to accurately calculate the
expansion of recurrence rules or recurrence overrides with the
utcStart property rather than the local start time. Even simple
recurrences such as "repeat weekly" may cross a daylight-savings
boundary and end up at a different UTC time. Clients that wish to
use "utcStart" are RECOMMENDED to request the server expand
recurrences (see Section XXX).

*__utcEnd__: UTCDate The server calculates the end time in UTC from
the start/timeZone/duration properties of the task. This is not
included by default and must be requested explicitly. Like
utcStart, this is calculated at fetch time if requested and may
change due to time zone data changes. Floating tasks will be
interpreted as per the time zone given as a Task/get argument.

5.1. Additional JSCalendar properties

This document defines four new JSCalendar properties.

5.1.1. mayInviteSelf

Type: Boolean (default: false)

If true, any user that has access to the event may add themselves to
it as a participant with the "attendee" role. This property MUST NOT
be altered in the recurrenceOverrides; it may only be set on the
master object.

5.1.2. mayInviteOthers

Type: Boolean (default: false)

If true, any current participant with the "attendee" role may add
new participants with the "attendee" role to the event. This
property MUST NOT be altered in the recurrenceOverrides; it may only
be set on the master object.

5.1.3. hideAttendees

Type: Boolean (default: false)

If true, only the owners of the event may see the full set of
participants. Other sharees of the event may only see the owners and
themselves. This property MUST NOT be altered in the
recurrenceOverrides; it may only be set on the master object.

5.1.4. relatedTo

Type: Id[String]|null (default: null)

A map of task ids to relations. Relation SHOULD be one of: -
blockedBy: Blocked by task with id. - clonedBy: Task with id was
cloned from this issue. - duplicatedBy: Task with id is a duplicate
of this issue. - causedBy: Task with id was the cause for this task.
- relatesTo: Task with id is related. - childOf: Task with id is
parent.

5.2. Properties similar in JMAP for Calendar

Attachments, per-user properties, recurrences and updates to
recurrences are described in [I-D.ietf-jmap-calendars], Section XXX.

5.3. Task/get

This is the "CalendarEvent/get" method as described in [I-D.ietf-
jmap-calendars], Section XXX.

TODO redefine this here. Similar to "TaskList/get" we only need to
replace a few definitions. For example, replace reduceParticipants
with reduceAssignees. Copy+Paste most of the stuff.

5.4. Task/changes

This is a standard "/changes" method as described in [RFC8620],
Section 5.2.

5.5. Task/set

This is the "CalendarEvent/set" method as described in [I-D.ietf-
jmap-calendars], Section XXX.

TODO copy+paste most stuff from "CalendarEvent/set". It should be
fine to just reference patching.

5.6. Task/copy

This is a standard "/copy" method as described in [RFC8620], Section 5.4.

5.7. Task/query

This is the "CalendarEvent/query" method as described in [I-D.ietf-jmap-calendars], Section XXX.

TODO copy+paste most stuff from "CalendarEvent/query". Mainly filtering should be different.

5.8. Task/queryChanges

This is a standard "/queryChanges" method as described in [RFC8620], Section 5.6.

6. Task Notifications

The TaskNotification data type records changes made by external entities to tasks in calendars the user is subscribed to. Notifications are stored in the same Account as the Task that was changed.

This is the same specification as the CalendarEventNotification object from [I-D.ietf-jmap-calendars], Section XXX. Only the object properties differ slightly and are therefore fully described in this document.

6.1. Object Properties

The **TaskNotification** object has the following properties:

* **id**: String The id of the TaskNotification.

* **created**: UTCDate The time this notification was created.

* **changedBy**: Person Who made the change.

  - **name**: String The name of the person who made the change.

  - **email**: String The email of the person who made the change, or null if no email is available.

  - **principalId**: String|null The id of the principal corresponding to the person who made the change, if any. This will be null if the change was due to receiving an iTIP message.

*   **comment**: String|null Comment sent along with the change by the
    user that made it. (e.g. COMMENT property in an iTIP message).

*   **type**: String This MUST be one of

    - created

    - updated

    - destroyed

*   **TaskId**: String The id of the Task that this notification is
    about.

*   **isDraft**: Boolean (created/updated only) Is this event a draft?

*   **event**: JSTask The data before the change (if updated or
    destroyed), or the data after creation (if created).

*   **eventPatch**: PatchObject (updated only) A patch encoding the
    change between the data in the event property, and the data after
    the update.

To reduce data, if the change only affects a single instance of a
recurring event, the server MAY set the event and eventPatch
properties for the instance; the calendarEventId MUST still be for
the master event.

6.2. TaskNotification/get

This is a standard "/get" method as described in [RFC8620], Section
5.1.

6.3. TaskNotification/changes

This is a standard "/changes" method as described in [RFC8620],
Section 5.2.

6.4. TaskNotification/set

This is a standard "/changes" method as described in [RFC8620],
Section 5.3.

Only destroy is supported; any attempt to create/update MUST be
rejected with a forbidden SetError.

6.5. TaskNotification/query

This is a standard "/query" method as described in [RFC8620],
Section 5.5.

6.5.1. Filtering

A **FilterCondition** object has the following properties:

* **after**: UTCDate|null The creation date must be on or after this date to match the condition.

* **before**: UTCDate|null The creation date must be before this date to match the condition.

* **type**: String The type property must be the same to match the condition.

* **taskIds**: Id[]|null A list of task ids. The taskId property of the notification must be in this list to match the condition.

6.5.2. Sorting

The "created" property MUST be supported for sorting.

6.6. TaskNotification/queryChanges

This is a standard "/queryChanges" method as described in [RFC8620], Section 5.6.

7. Security Considerations

All security considerations of JMAP for Calendars [I-D.ietf-jmap-calendars] apply to this specification.

8. IANA Considerations

8.1. JMAP Capability Registration for "tasks"

TODO Actually register

IANA will register the "tasks" JMAP Capability as follows:

Capability Name: urn:ietf:params:jmap:tasks

Specification document: this document

Intended use: common

Change Controller: IETF

Security and privacy considerations: this document, Section XXX

8.2. JSCalendar Property Registrations

All IANA registrations for JSTask are described in JMAP for
Calendars [I-D.ietf-jmap-calendars].

9. Normative References

[I-D.ietf-calext-jscalendar]
Jenkins, N. and R. Stepanek, "JSCalendar: A JSON
representation of calendar data", Work in Progress,
Internet-Draft, draft-ietf-calext-jscalendar-32, 15
October 2020, <https://tools.ietf.org/html/draft-ietf-
calext-jscalendar-32>.

[I-D.ietf-jmap-calendars]
Jenkins, N. and M. Douglass, "JMAP for Calendars", Work
in Progress, Internet-Draft, draft-ietf-jmap-
calendars-05, 24 January 2021, <https://tools.ietf.org/
html/draft-ietf-jmap-calendars-05>.

[I-D.jenkins-jmap-sharing]
Jenkins, N., "JMAP Sharing", Work in Progress, Internet-
Draft, draft-jenkins-jmap-sharing-00, 15 December 2020,
<https://tools.ietf.org/html/draft-jenkins-jmap-
sharing-00>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/
RFC2119, March 1997, <https://www.rfc-editor.org/info/
rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8620] Jenkins, N. and C. Newman, "The JSON Meta Application
Protocol (JMAP)", RFC 8620, DOI 10.17487/RFC8620, July
2019, <https://www.rfc-editor.org/info/rfc8620>.

10. Informative References

[RFC4791] Daboo, C., Desruisseaux, B., and L. Dusseault,
"Calendaring Extensions to WebDAV (CalDAV)", RFC 4791,
DOI 10.17487/RFC4791, March 2007, <https://www.rfc-
editor.org/info/rfc4791>.

Authors' Addresses

Joris Baum (editor)
audriga

Durlacher Allee 47

76131 Karlsruhe

Germany

Email: joris@audriga.com

URI: https://www.audriga.com

Hans-Joerg (editor)

audriga

Durlacher Allee 47

76131 Karlsruhe

Germany

Email: hans-joerg@audriga.com

URI: https://www.audriga.com