Onspring

PEOPLE-POWERED AUTOMATION®

# v2 Application Programming Interface (API)

## Administrator Guide

**Revised:** June 2024

# Contents

**v2 API Introduction** ..... **2**

v2 API Additional Capabilities ..... 2

**Generating an API Key** ..... **3**

**v2 Software Development Kit (SDK)** ..... **4**

**v2 API Request Overview** ..... **4**

Filter Options ..... 6

Handling Errors ..... 8

**v2 API FAQs** ..... **8**

# v2 API Introduction

Onspring provides a REST API that enables external programs to retrieve, save, and delete data within your Onspring instance. Onspring's v2 API implements version 3 of the Open API Specification (OAS).

**IMPORTANT:**

- Before data can be retrieved from or pushed into Onspring, you will need to create an API key. You must also assign this API key to a role that has access to the appropriate apps within Onspring. (See "Generating an API Key" below.)
- Onspring stores date/time information in **UTC (Coordinated Universal Time)**, and the API expects to pass UTC data back and forth. The Onspring GUI (graphical user interface) converts the UTC value to the appropriate display value depending on your instance configuration and user profile settings. If your source data is stored in a local time zone, you should convert it to UTC before passing the data to Onspring.
- For clients who program in .NET, you can access the v2 API SDK at github.com/onspring-technologies. If you are not using a .NET language, you can still interact with our API using HTTP endpoints and the features or libraries of your chosen language.
- If you wish to migrate an existing integration that uses the v1 API to the v2 API, please see the migration guide, located at github.com/onspring-technologies/onspring-api-sdk/blob/576ea9865f76d2e9150cd24a721b05f1e894ac9d/docs/migrations/2-to-30.md.

Visit **api.onspring.com/swagger** for supported GET, POST, PUT, and DELETE requests. Onspring's OAS metadata is available at api.onspring.com/swagger/v2/swagger.json. This JSON file may be used to generate client code to consume our API.

All interaction with the API occurs using the following base URL: **https://api.onspring.com**.

## v2 API Additional Capabilities

Onspring's v2 offers many capabilities beyond those found with the v1 API. All new integrations should utilize the v2 API to take advantage of the following:

- **Paging:** The v2 API allows you to retrieve a set number of records with each call rather than retrieving all records at once. The default number of records per page is 50, but this value may be adjusted up to 1000 records per page.

  **NOTE:** The total response is limited to 16MB, but the number of records that it takes to reach that limit is dependent on field count, reference size, data size per field, etc.
- **Batch Endpoints:** The v2 API allows you to retrieve batches of apps, fields, and records and to delete batches of records in a single call.
- **Improved Filtering:** The v2 API filter syntax supports Contains Any, Is Null and Not Null, helping you locate the data you need with greater precision. (See "Filter Options" below.)
- **List Value Interactions:** The v2 API allows you to add, edit, or remove values in shared lists and field-specific lists.

- **Attachment and Image Interactions:** The v2 API allows you to retrieve, add and remove files from attachment and image fields. You cannot modify the contents of files, but you can replace them.
- **Versioning:** With the v1 API, the version number was in the route of the endpoint you were calling. With the v2 API, the version is a header value. As the API continues to evolve and new versions are introduced, the routes will not need to be updated.

## Generating an API Key

In order to successfully interact with the v2 API, the external program must provide an X-ApiKey header in each HTTP request, using the following format:

```
X-ApiKey: 000000ffffff000000ffffff/00000000-ffff-0000-ffff-000000000000
```

A few important notes about API keys:

- The API key must be in an "Enabled" status.
- Each API key has an associated role that controls the permissions for requests made through the API key. The role must be in an "Active" status. If the API key does not have sufficient permissions to perform the requested action, an error will be returned. Onspring administrators may configure roles for API keys just as they would any other role in Onspring.
- Each API key can have only one assigned role.

To create an API key in Onspring, follow these steps:

1. Click the ⚙ icon in the navigation menu to open the **Administration** panel.
2. Click **Security** and then click **API Keys**.
3. Click **Create API Key**.
4. Select to **Add Blank API Key** or **Copy from**. If creating new, provide a unique name. If copying, select the API key you want to copy and provide a unique name.
5. Click **Save**.
6. Enter a **Description** for the API key, indicating its purpose and associated access permissions.
7. In the **Role** dropdown, select a role to assign to the API key.
8. Click the **Developer Information** tab.
9. Copy the **X-ApiKey Header** value.
10. Click **Save & Close**.

## v2 Software Development Kit (SDK)

If you are using .NET 4.6.1 or higher, you may want to use our SDK to interact with the Onspring v2 API. The SDK provides helpers that perform such tasks as formatting the calls to the API, returning strongly typed objects, and parsing errors. The SDK is located at:

**github.com/onspring-technologies/onspring-api-sdk**

The SDK can be included in a Visual Studio project via a Nuget package. This package is located at:

**nuget.org/packages/onspring.api.sdk**

## v2 API Request Overview

The requests that may be made to the v2 API are listed below. For requests that return data (except attachment or image files), the response body will be in JSON format. For additional details on supported requests, visit api.onspring.com/swagger.

| APPS | Request Purpose |
|---|---|
| `GET /Apps` | Gets all apps for the current client. |
| `GET /Apps/id/{appId}` | Gets an app by its identifier, if it exists, for the current client. |
| `POST /Apps/batch-get` | Gets up to 100 apps by their identifiers, for the current client. |

| FIELDS | Request Purpose |
|---|---|
| `GET /Fields/id/{fieldId}` | Gets the field by its identifier. |
| `POST /Fields/batch-get` | Gets a batch (max 100) of fields by their identifiers. |
| `GET /Fields/appId/{appId}` | Gets a list of fields for a given application. |

| FILES | Request Purpose |
|---|---|
| GET /Files/recordId/{recordId}/fieldId/{fieldId}/fileId/{fileId} | Gets a file's information. |
| DELETE /Files/recordId/{recordId}/fieldId/{fieldId}/fileId/{fileId} | Deletes a file attachment. |
| GET /Files/recordId/{recordId}/fieldId/{fieldId}/fileId/{fileId}/file | Gets a file's content. |
| POST /Files | Saves a file. |

| LISTS | Request Purpose |
|---|---|
| PUT /Lists/id/{listId}/items | Inserts/updates a list item. |
| DELETE /Lists/id/{listId}/itemId/{itemId} | Removes the specific item from the list. |

| PING | Request Purpose |
|---|---|
| GET /Ping | Gets a response indicating if the API is up or not. |

| RECORDS | Request Purpose |
|---|---|
| GET /Records/appId/{appId} | Gets a collection of records for a given app. |
| GET /Records/appId/{appId}/recordId/{recordId} | Gets a record by its identifier. |
| DELETE /Records/appId/{appId}/recordId/{recordId} | Deletes a record. |
| POST /Records/batch-get | Gets a batch of records. |
| POST /Records/Query | Queries the various records. (See “Filter Options” below.) |
| PUT /Records | Saves the provided record. If an Id is provided, we'll update it, otherwise a new record will be created. |

| RECORDS | Request Purpose |
|---|---|
| `POST /Records/batch-delete` | Deletes a batch of records for a given app. Only deletes records the user has access to. |

| REPORTS | Request Purpose |
|---|---|
| `GET /Reports/id/{reportId}` | Gets the report for the provided reportId. |
| `GET /Reports/appId/{appId}` | Gets the reports for the provided appId. |

## Filter Options

When using **POST /Records/Query**, please note the following filter options:

**eq** and **ne** –

- **Functionality:** Determines if the field equals or does not equal a value.
- **Applies to:** Text, number, date, and auto-number (Record Id) fields, along with formulas that have text, number, and date outputs
- **Examples:**
  - 221 eq 'High'
  - 95 ne 10

**contains** –

- **Functionality:** Determines if a list field contains the value. Value provided can be the list item identifier or the name of the list item.
- **Applies to:** List fields and formulas with list value outputs
- **Examples:**
  - 116 contains 'Good'
  - 116 contains 'cc8684ac-940d-4c51-9dc6-e276cfe63b57'

**isnull** and **notnull** –

- **Functionality:** Determines if the field contains or does not contain a null value.
- **Applies to:** Text, number, and date fields, along with formulas that have text, number, and date outputs
- **Examples:**
  - 68 isnull
  - 68 notnull

**lt** and **gt** –

- **Functionality:** Determines if a field is less than or greater than a value.
- **Applies to:** Number, date, and auto-number (Record Id) fields, along with formulas that have number and date outputs
- **Examples:**
  - 38 lt 1500
  - 38 gt 10

**Value Delimiters** –

- String values must be delimited with single quotation marks
- Number values require no delimiters
- Date values must be delimited with single quotation marks and preceded by the word datetime

**Component Expressions** – The and, not, or and parentheses may be used.

- **Examples:**
  - not (38 gt 10 or 36 eq 'Smith')
  - 37 gt datetime'2022-03-01T00:00:00.0000000'

**dataFormat** – Certain field types return different formats of data depending on which of the two values below is provided.

- **Raw** – The default if no value is provided.
  - AutoNumber fields (Record Id) return an integer value
  - Date fields return a date/time value
  - List fields return a Guid (if single-select) or an array of Guids (if multi-select) representing the selected list value ids
  - Number fields return a decimal value
  - Multi-line text fields return a string value that includes any html tags
  - Time span fields return an object with the following members:
    - Quantity – an integer representing the number of increments
    - Increment –
      - 2 for Seconds
      - 4 for Minutes
      - 8 for Hours
      - 16 for Days
      - 32 for Weeks
      - 64 for Months
      - 128 for Years
  - Recurrence –
    - 0 for None

        - 1 for EndByDate
        - 2 for EndAfterOccurrences
    - EndByDate – a date/time value used when Recurrence is 1
    - EndAfterOccurrences – an integer used when Recurrence is 2
- **Formatted** – Intended to be easier for people to read. For the field types described in Raw, returns a string value as follows:
    - AutoNumber fields (Record Id) apply the configured formatting
    - Date fields apply the configured formatting
    - List fields return the name (if single select) or an array of names (if multi-select) for each selected list value
    - Number fields apply the configured formatting
    - Multi-line text fields return a string value from which html tags have been removed
    - Time span fields return the string representation of the time span members (e.g., "Every 10 Day(s) End By 1/1/2022 6:00 AM")

## Handling Errors

Any request to the API may fail for various reasons related to API key permissions or the format of the request or its data. In these cases, the response will contain the following:

**One of these HTTP status codes:**

- 400 (Bad Request)
- 401 (Unauthorized)
- 403 (Forbidden)
- 404 (Not Found)

**A body in this format:**

```
{
    "Message":"<message>"
}
```

Where <message> may be either a simple string value or a serialized JSON value in this form:

```
{\"Errors\":[\"<message>\",\"<message>\"]}
```

# v2 API FAQs

**Is there a limit on the number of API calls I can make per day?**

Yes, there is a daily limit on API calls that is specified in your Onspring contract.

**Can I send my full data set or should I send only updated records?**

You should send only needed updates. This is part of how the daily request limit can be maintained.

**Do I have to send values for an entire record via the API or just changes?**

When updating records via the API, you can update as many or as few fields as needed. Updating only one or two fields is not an issue as long as a record match is found.

### Are default values set in records created through the API?

No, default field values will not be set in records created through the API. Your POST request should include all field values you wish to set within newly created records.

### How do I route API calls to the appropriate instance (ex: Prod vs. Dev)?

The API key that you generate in Onspring encodes all of the information needed to route your calls to the appropriate Onspring instance. All interaction with the API occurs using the following base URL: https://api.onspring.com. Then the API key encodes information to route the request to the correct instance and to authorize that request.

### Can I use the same API key in my Dev and Prod instances?

No, you will need to generate an API key in each instance, and those keys will be unique. (This is how requests are routed to the correct instance.) You can use the same role for each API key, but the role must exist in both instances.

### Do I need a user account to process API requests?

You do not need an Onspring user account to process API calls. However, you do need an API key in your Onspring instance with an associated role that grants permissions to all the apps where you will be retrieving, adding, or updating records via the API. Please see the Understand API Keys and Understand Roles topics for more details.

### Why is the Record ID field value different from the system-generated ID value used by the API?

Every record in Onspring has a stored Record ID that is a database-level system identifier. That is the ID that you see in the record URL, and it is used to manipulate records via the API. That ID encapsulates an entire record. In addition, every app in Onspring has an auto-number field that is called "Record Id" by default. (You can rename this field if desired.) On record creation, Onspring makes an effort to align the back-end system ID and the Record Id field value, but because of the way that Onspring queues background worker tasks, it isn't guaranteed that the values will be allocated together. When retrieving reports in the v2 API, the database-level Record ID value will be included for each record.

### How should I pass date/time information via the API?

Onspring stores date/time information in UTC (Coordinated Universal Time), and the API expects to pass UTC data back and forth. The Onspring GUI (graphical user interface) converts the UTC value to the appropriate display value depending on your instance configuration and user account settings. If your source data is stored in a local time zone, you should convert it to UTC before passing the data to Onspring.