
Ansible Tower API Guide

Release Ansible Tower 2.4.4

Red Hat, Inc.

Jun 06, 2017

CONTENTS

1	Introduction to the Tower API	2
1.1	Tools	2
1.2	Browsable API	2
1.3	Conventions	5
1.4	Sorting	6
1.5	Searching	6
1.6	Filtering	6
1.7	Pagination	8
1.8	Read-only Fields	9
1.9	Introduction to tower-cli	9
2	Auth Token API Endpoint	12
2.1	Reviewing the Options Endpoint	13
3	Ping API Endpoint	14
3.1	Reviewing the Options Endpoint	14
4	Configuration API Endpoint	15
4.1	Reviewing the Options Endpoint	16
5	Me API Endpoint	17
5.1	Reviewing the Options Endpoint	18
6	Dashboard API Endpoint	19
6.1	Reviewing the Options Endpoint	19
7	Organizations List API Endpoint	20
7.1	Results	21
7.2	Create Organizations	21
7.3	Reviewing the Options Endpoint	22
8	User List API Endpoint	23
8.1	Results	24
8.2	Create Users	24
8.3	Reviewing the Options Endpoint	25
9	Project List API Endpoint	26
9.1	Results	26
9.2	Create Projects	28
9.3	Reviewing the Options Endpoint	28

10 Team List API Endpoint	29
10.1 Results	29
10.2 Create Teams	30
10.3 Reviewing the Options Endpoint	30
11 Credential List API Endpoint	31
11.1 Results	31
11.2 Create Credentials	32
11.3 Reviewing the Options Endpoint	33
12 Inventory List API Endpoint	34
12.1 Results	34
12.2 Create Inventories	35
12.3 Reviewing the Options Endpoint	35
13 Inventory Script List API Endpoint	36
13.1 Results	36
13.2 Create Custom Inventory Scripts	37
13.3 Reviewing the Options Endpoint	37
14 Inventory Source List API Endpoint	38
14.1 Results	38
14.2 Reviewing the Options Endpoint	40
15 Group List API Endpoint	41
15.1 Results	41
15.2 Create Groups	42
15.3 Reviewing the Options Endpoint	42
16 Host List API Endpoint	43
16.1 Results	43
16.2 Create Hosts	44
16.3 Reviewing the Options Endpoint	44
17 Job Template List API Endpoint	45
17.1 Results	45
17.2 Create Job Templates	47
17.3 Reviewing the Options Endpoint	48
18 Job List API Endpoint	49
18.1 Results	49
18.2 Create Jobs	51
18.3 Reviewing the Options Endpoint	52
19 Ad Hoc Command List API Endpoint	53
19.1 Results	53
19.2 Create Ad Hoc Commands	55
19.3 Reviewing the Options Endpoint	56
20 System Job Template List API Endpoint	57
20.1 Results	57
20.2 Reviewing the Options Endpoint	58
21 System Job List API Endpoint	59
21.1 Results	59
21.2 Create System Jobs	60

21.3	Reviewing the Options Endpoint	61
22	Schedules API Endpoint	62
22.1	Results	62
22.2	Reviewing the Options Endpoint	63
23	Unified Job Template List API Endpoint	64
23.1	Results	64
23.2	Reviewing the Options Endpoint	65
24	Unified Job List API Endpoint	66
24.1	Results	66
24.2	Reviewing the Options Endpoint	67
25	Activity Stream List API Endpoint	68
25.1	Results	68
25.2	Reviewing the Options Endpoint	69
26	Index	70
27	Copyright © 2016 Red Hat, Inc.	71
	Index	72

Thank you for your interest in Ansible Tower by Red Hat, the open source IT orchestration engine. Whether sharing operations tasks with your team or integrating with Ansible through the Tower REST API, Tower provides many powerful tools to make your automation life easier.

The *Ansible Tower API Guide* focuses on helping you understand the Ansible Tower API. This document has been updated to include information for the latest release of Ansible Tower 2.4.4.

Ansible Tower Version 2.4.4; February 22, 2016; <https://access.redhat.com/>

INTRODUCTION TO THE TOWER API

1.1 Tools

This document offers a basic understanding of the REST API used by Ansible Tower.

REST stands for Representational State Transfer and is sometimes spelled as “ReST”. It relies on a stateless, client-server, and cacheable communications protocol, usually the HTTP protocol.

You may find it helpful see which API calls Tower makes in sequence. To do this, you can use the UI from Firebug or Chrome with developer plugins.

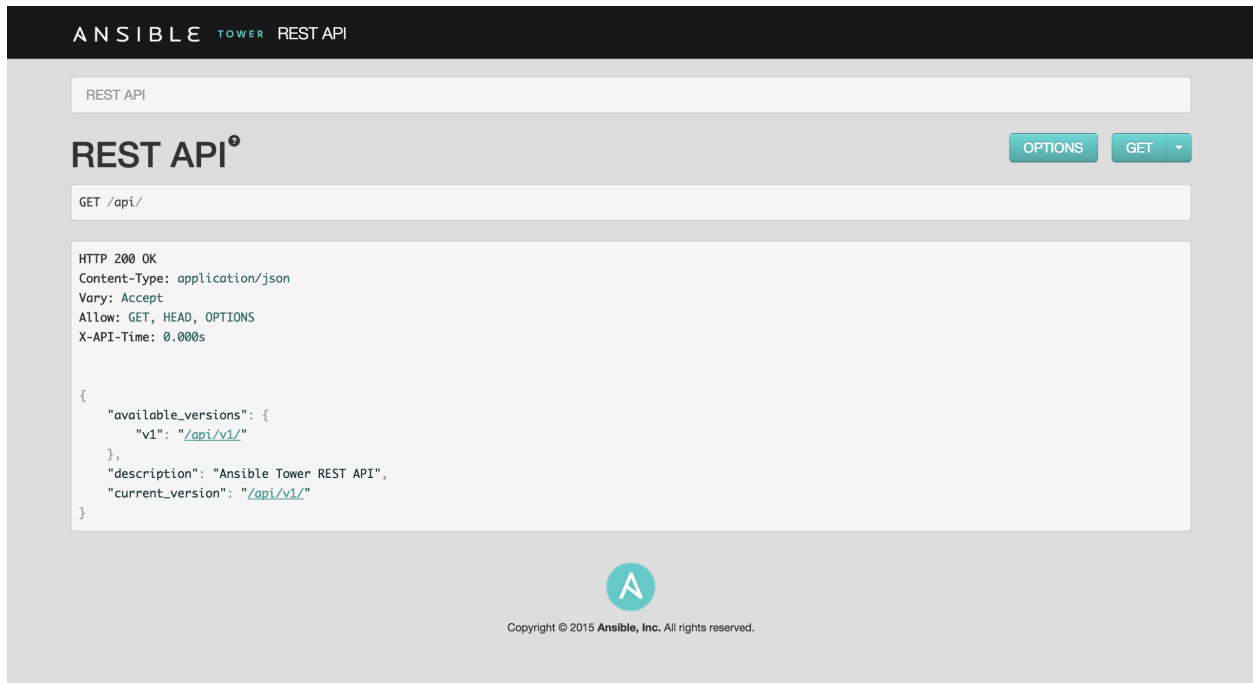
Another alternative is Charles Proxy (<http://www.charlesproxy.com/>), which offers a visualizer that you may find helpful. While it is commercial software, it can insert itself as an OS X proxy, for example, and intercept both requests from web browsers as well as curl and other API consumers.

Other alternatives include:

- Fiddler (<http://www.telerik.com/fiddler>)
- mitmproxy (<https://mitmproxy.org/>)
- Live HTTP headers FireFox extension (<https://addons.mozilla.org/en-US/firefox/addon/live-http-headers/>)
- Paros (<http://sourceforge.net/projects/paros/>)

1.2 Browsable API

REST APIs provide access to resources (data entities) via URI paths. You can visit the Ansible Tower REST API in a web browser at: `http://<Tower server name>/api/`



The screenshot displays the Ansible Tower REST API interface. At the top, the header reads "ANSIBLE TOWER REST API". Below this, there is a search bar containing "REST API". The main content area is titled "REST API" and includes a "GET /api/" request. The response is shown in a code block, displaying the following details:

```
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, HEAD, OPTIONS
X-API-Time: 0.000s

{
  "available_versions": {
    "v1": "/api/v1/"
  },
  "description": "Ansible Tower REST API",
  "current_version": "/api/v1/"
}
```

At the bottom of the interface, there is an Ansible logo and the copyright notice: "Copyright © 2015 Ansible, Inc. All rights reserved."

Clicking on various links in the API allows you to explore related resources.

ANSIBLE TOWER REST API

REST API · Version 1

Version 1 [?]

OPTIONS
GET ▼


GET /api/v1/

```


HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, HEAD, OPTIONS
X-API-Time: 0.004s

{
  "authtoken": "/api/v1/authtoken/",
  "ping": "/api/v1/ping/",
  "config": "/api/v1/config/",
  "me": "/api/v1/me/",
  "dashboard": "/api/v1/dashboard/",
  "organizations": "/api/v1/organizations/",
  "users": "/api/v1/users/",
  "projects": "/api/v1/projects/",
  "teams": "/api/v1/teams/",
  "credentials": "/api/v1/credentials/",
  "inventory": "/api/v1/inventories/",
  "inventory_scripts": "/api/v1/inventory_scripts/",
  "inventory_sources": "/api/v1/inventory_sources/",
  "groups": "/api/v1/groups/",
  "hosts": "/api/v1/hosts/",
  "job_templates": "/api/v1/job_templates/",
  "jobs": "/api/v1/jobs/",
  "ad_hoc_commands": "/api/v1/ad_hoc_commands/",
  "system_job_templates": "/api/v1/system_job_templates/",
  "system_jobs": "/api/v1/system_jobs/",
  "schedules": "/api/v1/schedules/",
  "unified_job_templates": "/api/v1/unified_job_templates/",
  "unified_jobs": "/api/v1/unified_jobs/",
  "activity_stream": "/api/v1/activity_stream/"
}

```



Copyright © 2015 Ansible, Inc. All rights reserved.

Clicking on the  next to the page name (toward the top of the screen) for an API endpoint gives you documentation on the access methods for that particular API endpoint and what data is returned when using those methods.

ANSIBLE TOWER REST API admin

REST API > Version 1 > Job List

Job List^o

OPTIONS GET

List Jobs:

Make a GET request to this resource to retrieve the list of jobs.

The resulting data structure contains:

```
{
  "count": 99,
  "next": null,
  "previous": null,
  "results": [
    ...
  ]
}
```

The `count` field indicates the total number of jobs found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more job records.

Results

Each job data structure includes the following fields:

- `id` : Database ID for this job. (integer, read-only)
- `type` : Data type for this job. (string, read-only)
- `url` : URL for this job. (string, read-only)
- `related` : Data structure with URLs of related resources. (object, read-only)
- `summary_fields` : Data structure with name/description for related resources. (object, read-only)
- `created` : Timestamp when this job was created. (datetime, read-only)
- `modified` : Timestamp when this job was last modified. (datetime, read-only)
- `name` : (string, required)
- `description` : (string)
- `unified_job_template` : (field)

You can also use PUT and POST verbs on the specific API pages by formatting JSON in the various text fields.

Media type

Content

```
{
  "name": "",
  "description": ""
}
```

POST

1.3 Conventions

Tower uses a standard REST API, rooted at `/api/` on the server. The API is versioned for compatibility reasons, but only `/api/v1/` is currently available. You can see information about what API versions are available by querying `/api/`.

You may have to specify the content/type on POST or PUT requests accordingly.

- PUT: Update a specific resource (by an identifier) or a collection of resources. PUT can also be used to create a specific resource if the resource identifier is known before-hand.
- POST: Create a new resource. Also acts as a catch-all verb for operations that do not fit into the other categories.

All URIs not ending with "/" receive a 301 redirect.

Note: Ansible Tower 2.4.4 API change: Formatting of `extra_vars` attached to Job Template records is preserved. Previously, YAML would be converted to JSON and returned as JSON. In 2.2.0 and newer, YAML is returned as YAML with formatting and comments preserved, and JSON is returned as JSON.

1.4 Sorting

To provide examples that are easy to follow, the following URL is used throughout this guide:

```
http://<Tower server name>/api/v1/groups/
```

To specify that `{{ model_verbose_name_plural }}` are returned in a particular order, use the `order_by` query string parameter on the GET request.

```
http://<Tower server name>/api/v1/model_verbose_name_plural?order_by={{ order_field }}
```

Prefix the field name with a dash (-) to sort in reverse:

```
http://<Tower server name>/api/v1/model_verbose_name_plural?order_by=-{{ order_field }}  
↔ }
```

Multiple sorting fields may be specified by separating the field names with a comma (,):

```
http://<Tower server name>/api/v1/model_verbose_name_plural?order_by={{ order_field }}  
↔ ,some_other_field
```

1.5 Searching

Use the search query string parameter to perform a case-insensitive search within all designated text fields of a model.

```
http://<Tower server name>/api/v1/model_verbose_name?search=findme
```

(Note: Added in AWX 1.4)

1.6 Filtering

Any collection is what the system calls a “queryset” and can be filtered via various operators.

For example, to find the groups that contain the name “foo”:

```
http://<Tower server name>/api/v1/groups/?name__contains=foo
```

To find an exact match:

```
http://<Tower server name>/api/v1/groups/?name=foo
```

If a resource is of an integer type, you must add `__int` to the end to cast your string input value to an integer, like so:

```
http://<Tower server name>/api/v1/arbitrary_resource/?x__int=5
```

Related resources can also be queried, like so:

```
http://<Tower server name>/api/v1/groups/?user__firstname__icontains=john
```

This will return all groups with users with names that include the string “John” in them.

You can also filter against multiple fields at once:

```
http://<Tower server name>/api/v1/groups/?user__firstname__icontains=john&group__name__icontains=foo
```

This finds all groups containing a user whose name contains “John” where the group contains the string foo.

For more about what types of operators are available, refer to:

<https://docs.djangoproject.com/en/dev/ref/models/querysets/>

Note: You can also watch the API as the UI is being used to see how it is filtering on various criteria.

Any additional query string parameters may be used to filter the list of results returned to those matching a given value. Only fields and relations that exist in the database may be used for filtering. Any special characters in the specified value should be url-encoded. For example:

```
?field=value%20xyz
```

Fields may also span relations, only for fields and relationships defined in the database:

```
?other__field=value
```

To exclude results matching certain criteria, prefix the field parameter with `not__`:

```
?not__field=value
```

(Added in AWX 1.4) By default, all query string filters are AND’ed together, so only the results matching all filters will be returned. To combine results matching any one of multiple criteria, prefix each query string parameter with `or__`:

```
?or__field=value&or__field=othervalue  
?or__not__field=value&or__field=othervalue
```

(Added in Ansible Tower 1.4.5) The default AND filtering applies all filters simultaneously to each related object being filtered across database relationships. The chain filter instead applies filters separately for each related object. To use, prefix the query string parameter with `chain__`:

```
?chain__related__field=value&chain__related__field2=othervalue  
?chain__not__related__field=value&chain__related__field2=othervalue
```

If the first query above were written as `?related__field=value&related__field2=othervalue`, it would return only the primary objects where the same related object satisfied both conditions. As written using the chain filter, it would return the intersection of primary objects matching each condition.

Field lookups may also be used for more advanced queries, by appending the lookup to the field name:

```
?field__lookup=value
```

The following field lookups are supported:

- `exact`: Exact match (default lookup if not specified).
- `iexact`: Case-insensitive version of `exact`.
- `contains`: Field contains value.
- `icontains`: Case-insensitive version of `contains`.
- `startswith`: Field starts with value.
- `istartswith`: Case-insensitive version of `startswith`.
- `endswith`: Field ends with value.
- `iendswith`: Case-insensitive version of `endswith`.
- `regex`: Field matches the given regular expression.
- `iregex`: Case-insensitive version of `regex`.
- `gt`: Greater than comparison.
- `gte`: Greater than or equal to comparison.
- `lt`: Less than comparison.
- `lte`: Less than or equal to comparison.
- `isnull`: Check whether the given field or related object is null; expects a boolean value.
- `in`: Check whether the given field's value is present in the list provided; expects a list of items.
- Boolean values may be specified as `True` or `1` for true, `False` or `0` for false (both case-insensitive).

Null values may be specified as `None` or `Null` (both case-insensitive), though it is preferred to use the `isnull` lookup to explicitly check for null values.

Lists (for the `in` lookup) may be specified as a comma-separated list of values.

1.7 Pagination

Responses for collections in the API are paginated. This means that while a collection may contain tens or hundreds of thousands of objects, in each web request, only a limited number of results are returned for API performance reasons.

When you get back the result for a collection you will see something similar to the following:

```
{'count': 25, 'next': 'http://testserver/api/v1/some_resource?page=2', 'previous': ↵
↵None, 'results': [ ... ] }
```

To get the next page, simply request the page given by the `'next'` sequential URL.

Use the `page_size=XX` query string parameter to change the number of results returned for each request.

Use the `page` query string parameter to retrieve a particular page of results.

```
http://<Tower server name>/api/v1/model_verbose_name?page_size=100&page=2
```

The previous and next links returned with the results will set these query string parameters automatically.

The serializer is quite efficient, but you should probably not request page sizes beyond a couple of hundred.

The user interface uses smaller values to avoid the user having to do a lot of scrolling.

1.8 Read-only Fields

Certain fields in the REST API are marked read-only. These usually include the URL of a resource, the ID, and occasionally some internal fields. For instance, the 'created_by' attribute of each object indicates which user created the resource, and cannot be edited.

If you post some values and notice that they are not changing, these fields may be read-only.

1.9 Introduction to tower-cli

tower-cli is a command line tool for Ansible Tower. It allows Tower commands to be easily run from the UNIX command line. It can also be used as a client library for other python apps, or as a reference for others developing API interactions with Tower's REST API.

Note: tower-cli is an open source project currently under development and, until a complete implementation occurs, only implements a subset of Tower's features.

1.9.1 License

While Tower is commercially licensed software, tower-cli is an open source project. Specifically, this project is licensed under the Apache 2.0 license. Pull requests, contributions, and tickets filed in GitHub are warmly welcomed.

1.9.2 Capabilities

tower-cli sends commands to the Tower API. It is capable of retrieving, creating, modifying, and deleting most objects within Tower.

A few potential uses include:

- Launching playbook runs (for instance, from Jenkins, TeamCity, Bamboo, etc)
- Checking on job statuses
- Rapidly creating objects like organizations, users, teams, and more

1.9.3 Installation

tower-cli is available as a package on [PyPI](#).

The preferred way to install is through pip:

```
$ pip install ansible-tower-cli
```

The main branch of this project may also be consumed directly from source.

For more information on tower-cli, refer to the project page at:

<https://github.com/ansible/tower-cli/>

1.9.4 Configuration

tower-cli can edit its own configuration or users can directly edit the configuration file, allowing configuration to be set in multiple ways.

Set configuration with tower-cli config

The preferred way to set configuration is with the tower-cli config command.

```
$ tower-cli config key value
```

By issuing the `tower-cli config` command without arguments, you can view a full list of configuration options and where they are set.

You will generally need to set at least three configuration options—`host`, `username`, and `password`—which correspond to the location of your Ansible Tower instance and your credentials to authenticate to Tower.

```
$ tower-cli config host tower.example.com
$ tower-cli config username leeroyjenkins
$ tower-cli config password myPassw0rd
```

Write to the config files directly.

The configuration file can also be edited directly. A configuration file is a simple file with keys and values, separated by `:` or `=`:

```
host: tower.example.com
username: admin
password: p4ssw0rd
```

File Locations

The order of precedence for configuration file locations is as follows, from least to greatest:

- internal defaults
- `/etc/awx/tower_cli.cfg` (written using `tower-cli config --global`)
- `~/.tower_cli.cfg` (written using `tower-cli config`)
- run-time parameters

Usage

CLI invocation generally follows this format:

```
$ tower-cli {resource} {action} ...
```

The “resource” is a type of object within Tower (a noun), such as `user`, `organization`, `job_template`, etc.; resource names are always singular in Tower CLI (use `tower-cli user`, never `tower-cli users`).

The “action” is the thing you want to do (a verb). Most Tower CLI resources have the following actions—`get`, `list`, `create`, `modify`, and `delete`—and have options corresponding to fields on the object in Tower.

Some examples:

```
# List all users.
$ tower-cli user list

# List all non-superusers
$ tower-cli user list --is-superuser=false

# Get the user with the ID of 42.
$ tower-cli user get 42

# Get the user with the given username.
$ tower-cli user get --username=guido

# Create a new user.
$ tower-cli user create --username=guido --first-name=Guido \
    --last-name="Van Rossum" --email=guido@python.org

# Modify an existing user.
# This would modify the first name of the user with the ID of "42" to "Guido".
$ tower-cli user modify 42 --first-name=Guido

# Modify an existing user, lookup by username.
# This would use "username" as the lookup, and modify the first name.
# Which fields are used as lookups vary by resource, but are generally
# the resource's name.
$ tower-cli user modify --username=guido --first-name=Guido

# Delete a user.
$ tower-cli user delete 42

# Launch a job.
$ tower-cli job launch --job-template=144

# Monitor a job.
$ tower-cli job monitor 95
```

When in doubt, help is available:

```
$ tower-cli # help
$ tower-cli user --help # resource specific help
$ tower-cli user create --help # command specific help
```

AUTH TOKEN API ENDPOINT

Make a POST request to this resource with `username` and `password` fields to obtain an authentication token to use for subsequent request.

```
GET /api/v1/authtoken/

HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: POST, OPTIONS
X-API-Time: 0.001s

{
  "detail": "Method 'GET' not allowed."
}
```

Example JSON to POST (content type is `application/json`):

```
{"username": "user", "password": "my pass"}
```

Example form data to post (content type is `application/x-www-form-urlencoded`):

```
username=user&password=my%20pass
```

If the username and password provided are valid, the response will contain a `token` field with the authentication token to use and an `expires` field with the timestamp when the token will expire:

```
{
  "token": "8f17825cf08a7efea124f2638f3896f6637f8745",
  "expires": "2013-09-05T21:46:35.729Z"
}
```

Otherwise, the response will indicate the error that occurred and return a 4xx status code.

For subsequent requests, pass the token via the HTTP `Authorization` request header:

```
Authorization: Token 8f17825cf08a7efea124f2638f3896f6637f8745
```

The auth token is only valid when used from the same remote address and user agent that originally obtained it.

Each request that uses the token for authentication will refresh its expiration timestamp and keep it from expiring. A token only expires when it is not used for the configured timeout interval (default 1800 seconds).

2.1 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

PING API ENDPOINT

A simple view that reports very basic information about this Tower instance, which is acceptable to be public information.

```
GET /api/v1/ping/
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, HEAD, OPTIONS
X-API-Time: 0.003s

{
  "instances": {
    "primary": "localhost",
    "secondaries": []
  },
  "ha": false,
  "role": "primary",
  "version": "2.2.2"
}
```

3.1 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

CONFIGURATION API ENDPOINT

Site configuration settings and general information.

```
GET /api/v1/config/
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, POST, DELETE, HEAD, OPTIONS
X-API-Time: 0.223s

{
  "eula": "TOWER SOFTWARE END USER LICENSE AGREEMENT Unless otherwise agreed to,
↪and executed in a definitive agreement, between Ansible, Inc.
  (201cAnsible201d) and the individual or entity (201cCustomer201d) signing or
↪electronically accepting these terms of use for the Tower Software
  (201cEULA201d), all Tower Software, including any and all versions released or
↪made available by Ansible, shall be subject to the Ansible Software
  Subscription and Services Agreement found at http://www.ansible.com/subscription-
↪agreement (201cAgreement201d). Ansible is not responsible for
  any additional obligations, conditions or warranties agreed to between Customer
↪and an authorized distributor, or reseller, of the Tower Software.
  BY DOWNLOADING AND USING THE TOWER SOFTWARE, OR BY CLICKING ON THE 201cYES201d
↪BUTTON OR OTHER BUTTON OR MECHANISM DESIGNED TO ACKNOWLEDGE CONSENT
  TO THE TERMS OF AN ELECTRONIC COPY OF THIS EULA, THE CUSTOMER HEREBY ACKNOWLEDGES
↪THAT CUSTOMER HAS READ, UNDERSTOOD, AND AGREES TO BE BOUND BY THE
  TERMS OF THIS EULA AND AGREEMENT, INCLUDING ALL TERMS INCORPORATED HEREIN BY
↪REFERENCE, AND THAT THIS EULA AND AGREEMENT IS EQUIVALENT TO ANY
  WRITTEN NEGOTIATED AGREEMENT BETWEEN CUSTOMER AND ANSIBLE. THIS EULA AND
↪AGREEMENT IS ENFORCEABLE AGAINST ANY PERSON OR ENTITY THAT USES OR AVAILS
  ITSELF OF THE TOWER SOFTWARE OR ANY PERSON OR ENTITY THAT USES THE OR AVAILS
↪ITSELF OF THE TOWER SOFTWARE ON ANOTHER PERSON2019S OR ENTITY2019S BEHALF.",
  "license_info": {},
  "analytics_status": "detailed",
  "version": "3.0.0-0.git201510071505",
  "project_base_dir": "/var/lib/awx/projects",
  "time_zone": "America/New_York",
  "ansible_version": "1.9.2",
  "project_local_paths": []
}
```

Make a GET request to this resource to retrieve the configuration containing the following fields (some fields may not be visible to all users):

- `project_base_dir`: Path on the server where projects and playbooks are stored.

- `project_local_paths`: List of directories beneath `project_base_dir` to use when creating/editing a project.
- `time_zone`: The configured time zone for the server.
- `license_info`: Information about the current license.
- `version`: Version of Ansible Tower package installed.
- `eula`: The current End-User License Agreement

(New in Ansible Tower 2.0.0) Make a POST request to this resource as a super user to install or update the existing license. The license data itself can be POSTed as a normal json data structure.

(New in Ansible Tower 2.1.1) The POST must include a `eula_accepted` boolean element indicating acceptance of the End-User License Agreement.

4.1 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

ME API ENDPOINT

Make a GET request to retrieve user information about the current user.

```
GET /api/v1/me/
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, HEAD, OPTIONS
X-API-Time: 0.055s

{
  "count": 1,
  "next": null,
  "previous": null,
  "results": [
    {
      "id": 1,
      "type": "user",
      "url": "/api/v1/users/1/",
      "related": {
        "admin_of_organizations": "/api/v1/users/1/admin_of_organizations/",
        "organizations": "/api/v1/users/1/organizations/",
        "credentials": "/api/v1/users/1/credentials/",
        "permissions": "/api/v1/users/1/permissions/",
        "activity_stream": "/api/v1/users/1/activity_stream/",
        "projects": "/api/v1/users/1/projects/",
        "teams": "/api/v1/users/1/teams/"
      },
      "created": "2015-08-25T01:00:07.127Z",
      "username": "admin",
      "first_name": "",
      "last_name": "",
      "email": "admin@example.com",
      "is_superuser": true,
      "ldap_dn": ""
    }
  ]
}
```

One result should be returned containing the following fields:

- `id`: Database ID for this user. (integer)
- `type`: Data type for this user. (multiple choice)
- `user`: User

- `url`: URL for this user. (string)
- `related`: Data structure with URLs of related resources. (object)
- `created`: Timestamp when this user was created. (datetime)
- `username`: Required. 30 characters or fewer. Letters, numbers and `@/./+/_` characters (string)
- `first_name`: (string)
- `last_name`: (string)
- `email`: (email)
- `is_superuser`: Designates that this user has all permissions without explicitly assigning them. (boolean)
- `password`: Write-only field used to change the password. (string)
- `ldap_dn`: (string)

Use the primary URL for the user (`/api/v1/users/N/`) to modify the user.

5.1 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

DASHBOARD API ENDPOINT

Make a GET request to this resource to retrieve aggregate statistics for Tower. (Added in AWX 1.4)

```
GET /api/v1/dashboard/
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, HEAD, OPTIONS
X-API-Time: 0.087s

{
  "name": "Dashboard",
  "description": "",
  "renders": [
    "application/json",
    "text/html"
  ],
  "parses": [
    "application/json"
  ],
  "added_in_version": "1.4"
}
```

6.1 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

ORGANIZATIONS LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of organizations.

```
GET /api/v1/organizations/
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, POST, HEAD, OPTIONS
X-API-Time: 0.055s
```

The resulting data structure contains:

```
{
  "count": 99,
  "next": null,
  "previous": null,
  "results": [
    ...
  ]
}
```

The `count` field indicates the total number of organizations found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more organization records.

```
{
  "count": 1,
  "next": null,
  "previous": null,
  "results": [
    {
      "id": 1,
      "type": "organization",
      "url": "/api/v1/organizations/1/",
      "related": {
        "created_by": "/api/v1/users/1/",
        "modified_by": "/api/v1/users/1/",
        "admins": "/api/v1/organizations/1/admins/",
        "users": "/api/v1/organizations/1/users/",
        "inventories": "/api/v1/organizations/1/inventories/",
        "activity_stream": "/api/v1/organizations/1/activity_stream/",
        "projects": "/api/v1/organizations/1/projects/",
        "teams": "/api/v1/organizations/1/teams/"
      },
      "summary_fields": {
```



```

        "created_by": {
            "id": 1,
            "username": "admin",
            "first_name": "",
            "last_name": ""
        },
        "modified_by": {
            "id": 1,
            "username": "admin",
            "first_name": "",
            "last_name": ""
        }
    },
    "created": "2015-08-25T01:00:18.545Z",
    "modified": "2015-08-25T01:00:18.545Z",
    "name": "Default",
    "description": ""
}
]
}

```

7.1 Results

Each organization data structure includes the following fields:

- `id`: Database ID for this organization. (integer)
- `type`: Data type for this organization. (multiple choice)
- `organization`: Organization
- `url`: URL for this organization. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `created`: Timestamp when this organization was created. (datetime)
- `modified`: Timestamp when this organization was last modified. (datetime)
- `name`: (string)
- `description`: (string)

7.2 Create Organizations

Make a POST request to this resource with the following organization fields to create a new organization:

- `name`: (string, required)
- `description`: (string, default="")

7.3 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

USER LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of users.

```
GET /api/v1/users/
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, POST, HEAD, OPTIONS
X-API-Time: 0.049s

{
  "count": 1,
  "next": null,
  "previous": null,
  "results": [
    {
      "id": 1,
      "type": "user",
      "url": "/api/v1/users/1/",
      "related": {
        "admin_of_organizations": "/api/v1/users/1/admin_of_organizations/",
        "organizations": "/api/v1/users/1/organizations/",
        "credentials": "/api/v1/users/1/credentials/",
        "permissions": "/api/v1/users/1/permissions/",
        "activity_stream": "/api/v1/users/1/activity_stream/",
        "projects": "/api/v1/users/1/projects/",
        "teams": "/api/v1/users/1/teams/"
      },
      "created": "2015-08-25T01:00:07.127Z",
      "username": "admin",
      "first_name": "",
      "last_name": "",
      "email": "admin@example.com",
      "is_superuser": true,
      "ldap_dn": ""
    }
  ]
}
```

The resulting data structure contains:

```
{
  "count": 99,
  "next": null,
  "previous": null,
```

```

"results": [
  ...
]
}

```

The `count` field indicates the total number of users found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more user records.

8.1 Results

Each user data structure includes the following fields:

- `id`: Database ID for this user. (integer)
- `type`: Data type for this user. (multiple choice)
- `user`: User
- `url`: URL for this user. (string)
- `related`: Data structure with URLs of related resources. (object)
- `created`: Timestamp when this user was created. (datetime)
- `username`: Required. 30 characters or fewer. Letters, numbers and `@/./+/_` characters (string)
- `first_name`: (string)
- `last_name`: (string)
- `email`: (email)
- `is_superuser`: Designates that this user has all permissions without explicitly assigning them. (boolean)
- `password`: Write-only field used to change the password. (string)
- `ldap_dn`: (string)

8.2 Create Users

Make a POST request to this resource with the following user fields to create a new user:

- `username`: Required. 30 characters or fewer. Letters, numbers and `@/./+/_` characters (string, required)
- `first_name`: (string, default="")
- `last_name`: (string, default="")
- `email`: (email, default="")
- `is_superuser`: Designates that this user has all permissions without explicitly assigning them. (boolean, default=False)
- `password`: Write-only field used to change the password. (string, default="")

8.3 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

PROJECT LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of projects.

```
GET /api/v1/projects/
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, POST, HEAD, OPTIONS
X-API-Time: 0.053s
```

The resulting data structure contains:

```
{
  "count": 99,
  "next": null,
  "previous": null,
  "results": [
    ...
  ]
}
```

The `count` field indicates the total number of projects found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more project records.

9.1 Results

Each project data structure includes the following fields:

- `id`: Database ID for this project. (integer)
- **type: Data type for this project. (multiple choice)**
 - `project`: Project
- `url`: URL for this project. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `created`: Timestamp when this project was created. (datetime)
- `modified`: Timestamp when this project was last modified. (datetime)
- `name`: (string)

- `description`: (string)
- `local_path`: Local path (relative to `PROJECTS_ROOT`) containing playbooks and related files for this project. (string)
- **`scm_type`: (multiple choice)**
 - `"`: Manual
 - `git`: Git
 - `hg`: Mercurial
 - `svn`: Subversion
- `scm_url`: (string)
- `scm_branch`: Specific branch, tag or commit to checkout. (string)
- `scm_clean`: (boolean)
- `scm_delete_on_update`: (boolean)
- `credential`: (field)
- `last_job_run`: (datetime)
- `last_job_failed`: (boolean)
- `has_schedules`: (boolean)
- `next_job_run`: (datetime)
- **`status`: (multiple choice)**
 - `new`: New
 - `pending`: Pending
 - `waiting`: Waiting
 - `running`: Running
 - `successful`: Successful
 - `failed`: Failed
 - `error`: Error
 - `canceled`: Canceled
 - `never updated`: Never Updated
 - `ok`: OK
 - `missing`: Missing
- `scm_delete_on_next_update`: (boolean)
- `scm_update_on_launch`: (boolean)
- `scm_update_cache_timeout`: (integer)
- `last_update_failed`: (boolean)
- `last_updated`: (datetime)

9.2 Create Projects

Make a POST request to this resource with the following project fields to create a new project:

- `name`: (string, required)
- `description`: (string, default="")
- `local_path`: Local path (relative to `PROJECTS_ROOT`) containing playbooks and related files for this project. (string, default="")
- **`scm_type`: (multiple choice)**
 - `"`: Manual (default)
 - `git`: Git
 - `hg`: Mercurial
 - `svn`: Subversion
- `scm_url`: (string, default="")
- `scm_branch`: Specific branch, tag or commit to checkout. (string, default="")
- `scm_clean`: (boolean, default=False)
- `scm_delete_on_update`: (boolean, default=False)
- `credential`: (field, default=None)
- `scm_update_on_launch`: (boolean, default=False)
- `scm_update_cache_timeout`: (integer, default=0)

9.3 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

TEAM LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of teams.

```
GET /api/v1/teams/  
HTTP 200 OK  
Content-Type: application/json  
Vary: Accept  
Allow: GET, POST, HEAD, OPTIONS  
X-API-Time: 0.046s
```

The resulting data structure contains:

```
{  
  "count": 0,  
  "next": null,  
  "previous": null,  
  "results": []  
}
```

The `count` field indicates the total number of teams found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more team records.

10.1 Results

Each team data structure includes the following fields:

- `id`: Database ID for this team. (integer)
- `type`: Data type for this team. (multiple choice)
- `team`: Team
- `url`: URL for this team. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `created`: Timestamp when this team was created. (datetime)
- `modified`: Timestamp when this team was last modified. (datetime)
- `name`: (string)
- `description`: (string)

- `organization`: (field)

10.2 Create Teams

Make a POST request to this resource with the following team fields to create a new team:

- `name`: (string, required)
- `description`: (string, default="")
- `organization`: (field, default=None)

10.3 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

CREDENTIAL LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of credentials.

```
GET /api/v1/credentials/
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, POST, HEAD, OPTIONS
X-API-Time: 0.046s

{
  "count": 0,
  "next": null,
  "previous": null,
  "results": []
}
```

The `count` field indicates the total number of hosts found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more credential records.

11.1 Results

Each credential data structure includes the following fields:

- `id`: Database ID for this credential. (integer)
- **type**: Data type for this credential. (multiple choice)
 - `credential`: Credential
- `url`: URL for this credential. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `created`: Timestamp when this credential was created. (datetime)
- `modified`: Timestamp when this credential was last modified. (datetime)
- `name`: (string)
- `description`: (string)
- `user`: (field)

- team: (field)
- **kind: (multiple choice)**
 - ssh: Machine
 - scm: Source Control
 - aws: Amazon Web Services
 - rax: Rackspace
 - vmware: VMware vCenter
 - gce: Google Compute Engine
 - azure: Microsoft Azure
 - openstack: OpenStack
- cloud: (boolean)
- host: The hostname or IP address to use. (string)
- username: Username for this credential. (string)
- password: (string)
- security token: (string)
- project: The identifier for the project. (string)
- ssh_key_data: (string)
- ssh_key_unlock: (string)
- **become_method: Privilege escalation method. (multiple choice)**
 - "": None
 - sudo: Sudo
 - su: Su
 - pbrun: Pbrun
 - pfexec: Pfexec
- become_username: Privilege escalation username. (string)
- become_password: (string)
- vault_password: (string)

11.2 Create Credentials

Make a POST request to this resource with the following credential fields to create a new credential:

- name: (string, required)
- description: (string, default="")
- user: (field, default=None)
- team: (field, default=None)
- **kind: (multiple choice, required)**

- ssh: Machine (default)
- scm: Source Control
- aws: Amazon Web Services
- rax: Rackspace
- vmware: VMware vCenter
- gce: Google Compute Engine
- azure: Microsoft Azure
- openstack: OpenStack
- host: The hostname or IP address to use. (string, default="")
- username: Username for this credential. (string, default="")
- password: (string, default="")
- security token: (string, default="")
- project: The identifier for the project. (string, default="")
- ssh_key_data: (string, default="")
- ssh_key_unlock: (string, default="")
- **become_method: Privilege escalation method. (multiple choice)**
 - "": None (default)
 - sudo: Sudo
 - su`: Su
 - pbrun: Pbrun
 - pfexec: Pfexec
- become_username: Privilege escalation username. (string, default="")
- become_password: (string, default="")
- vault_password: (string, default="")

11.3 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

INVENTORY LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of inventories.

```
GET /api/v1/inventories/  
HTTP 200 OK  
Content-Type: application/json  
Vary: Accept  
Allow: GET, POST, HEAD, OPTIONS  
X-API-Time: 0.046s
```

The resulting data structure contains:

```
{  
  "count": 0,  
  "next": null,  
  "previous": null,  
  "results": []  
}
```

The `count` field indicates the total number of inventory lists found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more inventory records.

12.1 Results

Each inventory data structure includes the following fields:

- `id`: Database ID for this inventory. (integer)
- **type: Data type for this inventory. (multiple choice)**
 - `inventory`: Inventory
- `url`: URL for this inventory. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `created`: Timestamp when this inventory was created. (datetime)
- `modified`: Timestamp when this inventory was last modified. (datetime)
- `name`: (string)
- `description`: (string)

- `organization`: (field)
- `variables`: Inventory variables in JSON or YAML format. (string)
- `has_active_failures`: Flag indicating whether any hosts in this inventory have failed. (boolean)
- `total_hosts`: Total number of hosts in this inventory. (integer)
- `hosts_with_active_failures`: Number of hosts in this inventory with active failures. (integer)
- `total_groups`: Total number of groups in this inventory. (integer)
- `groups_with_active_failures`: Number of groups in this inventory with active failures. (integer)
- `has_inventory_sources`: Flag indicating whether this inventory has any external inventory sources. (boolean)
- `total_inventory_sources`: Total number of external inventory sources configured within this inventory. (integer)
- `inventory_sources_with_failures`: Number of external inventory sources in this inventory with failures. (integer)

12.2 Create Inventories

Make a POST request to this resource with the following inventory fields to create a new inventory:

- `name`: (string, required)
- `description`: (string, default="")
- `organization`: (field, required)
- `variables`: Inventory variables in JSON or YAML format. (string, default="")

12.3 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

INVENTORY SCRIPT LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of custom inventory scripts.

```
GET /api/v1/inventory_scripts/  
HTTP 200 OK  
Content-Type: application/json  
Vary: Accept  
Allow: GET, POST, HEAD, OPTIONS  
X-API-Time: 0.049s
```

The resulting data structure contains:

```
{  
  "count": 0,  
  "next": null,  
  "previous": null,  
  "results": []  
}
```

The `count` field indicates the total number of inventories found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more inventory script records.

13.1 Results

Each custom inventory script data structure includes the following fields:

- `id`: Database ID for this custom inventory script. (integer)
- **type**: Data type for this custom inventory script. (multiple choice)
 - `custom_inventory_script`: Custom Inventory Script
- `url`: URL for this custom inventory script. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `created`: Timestamp when this custom inventory script was created. (datetime)
- `modified`: Timestamp when this custom inventory script was last modified. (datetime)
- `name`: (string)
- `description`: (string)

- `script`: Inventory script contents (string)
- `organization`: (field)

13.2 Create Custom Inventory Scripts

Make a POST request to this resource with the following custom inventory script fields to create a new custom inventory script:

- `name`: (string, required)
- `description`: (string, default="")
- `script`: Inventory script contents (string, default="")
- `organization`: (field, required)

13.3 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

INVENTORY SOURCE LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of inventory sources.

```
GET /api/v1/inventory_sources/  
HTTP 200 OK  
Content-Type: application/json  
Vary: Accept  
Allow: GET, HEAD, OPTIONS  
X-API-Time: 0.053s
```

The resulting data structure contains:

```
{  
  "count": 99,  
  "next": null,  
  "previous": null,  
  "results": []  
}
```

The `count` field indicates the total number of inventory sources found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more inventory source records.

14.1 Results

Each inventory source data structure includes the following fields:

- `id`: Database ID for this inventory source. (integer)
- **type: Data type for this inventory source. (multiple choice)**
 - `inventory_source`: Inventory Source
- `url`: URL for this inventory source. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `created`: Timestamp when this inventory source was created. (datetime)
- `modified`: Timestamp when this inventory source was last modified. (datetime)
- `name`: (string)
- `description`: (string)

- **source: (multiple choice)**
 - "": Manual
 - file: Local File, Directory or Script
 - rax: Rackspace Cloud Servers
 - ec2: Amazon EC2
 - gce: Google Compute Engine
 - azure: Microsoft Azure
 - vmware: VMware vCenter
 - openstack: OpenStack
 - custom: Custom Script
- source_path: (string)
- source_script: (field)
- source_vars: Inventory source variables in YAML or JSON format. (string)
- credential: (field)
- source_regions: (string)
- instance_filters: Comma-separated list of filter expressions (EC2 only). Hosts are imported when ANY of the filters match. (string)
- group_by: Limit groups automatically created from inventory source (EC2 only). (string)
- overwrite: Overwrite local groups and hosts from remote inventory source. (boolean)
- overwrite_vars: Overwrite local variables from remote inventory source. (boolean)
- last_job_run: (datetime)
- last_job_failed: (boolean)
- has_schedules: (boolean)
- next_job_run: (datetime)
- **status: (multiple choice)**
 - new: New
 - pending: Pending
 - waiting: Waiting
 - running: Running
 - successful: Successful
 - failed: Failed
 - error: Error
 - canceled: Canceled
 - never updated: Never Updated
 - none: No External Source
- inventory: (field)

- `group`: (field)
- `update_on_launch`: (boolean)
- `update_cache_timeout`: (integer)
- `last_update_failed`: (boolean)
- `last_updated`: (datetime)

14.2 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

GROUP LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of groups.

```
GET /api/v1/groups/
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, POST, HEAD, OPTIONS
X-API-Time: 0.049s
```

The resulting data structure contains:

```
{
  "count": 99,
  "next": null,
  "previous": null,
  "results": [
    ...
  ]
}
```

The `count` field indicates the total number of groups found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more group records.

15.1 Results

Each group data structure includes the following fields:

- `id`: Database ID for this group. (integer)
- **type: Data type for this group. (multiple choice)**
 - `group`: Group
- `url`: URL for this group. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `created`: Timestamp when this group was created. (datetime)
- `modified`: Timestamp when this group was last modified. (datetime)
- `name`: (string)

- `description`: (string)
- `inventory`: (field)
- `variables`: Group variables in JSON or YAML format. (string)
- `has_active_failures`: Flag indicating whether this group has any hosts with active failures. (boolean)
- `total_hosts`: Total number of hosts directly or indirectly in this group. (integer)
- `hosts_with_active_failures`: Number of hosts in this group with active failures. (integer)
- `total_groups`: Total number of child groups contained within this group. (integer)
- `groups_with_active_failures`: Number of child groups within this group that have active failures. (integer)
- `has_inventory_sources`: Flag indicating whether this group was created/updated from any external inventory sources. (boolean)

15.2 Create Groups

Make a POST request to this resource with the following group fields to create a new group:

- `name`: (string, required)
- `description`: (string, default="")
- `inventory`: (field, required)
- `variables`: Group variables in JSON or YAML format. (string, default="")

15.3 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

HOST LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of hosts.

```
GET /api/v1/hosts/
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, POST, HEAD, OPTIONS
X-API-Time: 0.050s
```

The resulting data structure contains:

```
{
  "count": 99,
  "next": null,
  "previous": null,
  "results": [
    ...
  ]
}
```

The `count` field indicates the total number of hosts found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more host records.

16.1 Results

Each host data structure includes the following fields:

- `id`: Database ID for this host. (integer)
- **type: Data type for this host. (multiple choice)**
 - `host`: Host
- `url`: URL for this host. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `created`: Timestamp when this host was created. (datetime)
- `modified`: Timestamp when this host was last modified. (datetime)
- `name`: (string)

- `description`: (string)
- `inventory`: (field)
- `enabled`: Is this host online and available for running jobs? (boolean)
- `instance_id`: (string)
- `variables`: Host variables in JSON or YAML format. (string)
- `has_active_failures`: Flag indicating whether the last job failed for this host. (boolean)
- `has_inventory_sources`: Flag indicating whether this host was created/updated from any external inventory sources. (boolean)
- `last_job`: (field)
- `last_job_host_summary`: (field)

16.2 Create Hosts

Make a POST request to this resource with the following host fields to create a new host:

- `name`: (string, required)
- `description`: (string, default="")
- `inventory`: (field, required)
- `enabled`: Is this host online and available for running jobs? (boolean, default=True)
- `instance_id`: (string, default="")
- `variables`: Host variables in JSON or YAML format. (string, default="")
- `last_job`: (field, default=None)
- `last_job_host_summary`: (field, default=None)

16.3 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

JOB TEMPLATE LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of job templates.

```
GET /api/v1/job_templates/  
HTTP 200 OK  
Content-Type: application/json  
Vary: Accept  
Allow: GET, POST, HEAD, OPTIONS  
X-API-Time: 0.051s
```

The resulting data structure contains:

```
{  
  "count": 99,  
  "next": null,  
  "previous": null,  
  "results": [  
    ...  
  ]  
}
```

The `count` field indicates the total number of job templates found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more job template records.

17.1 Results

Each job template data structure includes the following fields:

- `id`: Database ID for this job template. (integer)
- **type: Data type for this job template. (multiple choice)**
 - `job_template`: Job Template
- `url`: URL for this job template. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `created`: Timestamp when this job template was created. (datetime)
- `modified`: Timestamp when this job template was last modified. (datetime)
- `name`: (string)

- description: (string)
- **job_type: (multiple choice)**
 - run: Run
 - check: Check
 - scan: Scan
- inventory: (field)
- project: (field)
- playbook: (string)
- credential: (field)
- cloud_credential: (field)
- forks: (integer)
- limit: (string)
- **verbosity: (multiple choice)**
 - 0: 0 (Normal)
 - 1: 1 (Verbose)
 - 2: 2 (More Verbose)
 - 3: 3 (Debug)
 - 4: 4 (Connection Debug)
 - 5: 5 (WinRM Debug)
- extra_vars: (string)
- job_tags: (string)
- force_handlers: (boolean)
- skip_tags: (string)
- start_at_task: (string)
- last_job_run: (datetime)
- last_job_failed: (boolean)
- has_schedules: (boolean)
- next_job_run: (datetime)
- **status: (multiple choice)**
 - new: New
 - pending: Pending
 - waiting: Waiting
 - running: Running
 - successful: Successful
 - failed: Failed
 - error: Error

- canceled: Canceled
- never updated: Never Updated
- host_config_key: (string)
- ask_variables_on_launch: (boolean)
- survey_enabled: (boolean)
- become_enabled: (boolean)

17.2 Create Job Templates

Make a POST request to this resource with the following job template fields to create a new job template:

- name: (string, required)
- description: (string, default="")
- **job_type: (multiple choice, required)**
 - run: Run (default)
 - check: Check
 - scan: Scan
- inventory: (field, default=None)
- project: (field, default=None)
- playbook: (string, default="")
- credential: (field, default=None)
- cloud_credential: (field, default=None)
- forks: (integer, default=0)
- limit: (string, default="")
- **verbosity: (multiple choice)**
 - 0: 0 (Normal) (default)
 - 1: 1 (Verbose)
 - 2: 2 (More Verbose)
 - 3: 3 (Debug)
 - 4: 4 (Connection Debug)
 - 5: 5 (WinRM Debug)
- extra_vars: (string, default="")
- job_tags: (string, default="")
- force_handlers: (boolean, default=False)
- skip_tags: (string, default="")
- start_at_task: (string, default="")
- host_config_key: (string, default="")

- `ask_variables_on_launch`: (boolean, default=False)
- `survey_enabled`: (boolean, default=False)
- `become_enabled`: (boolean, default=False)

17.3 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

JOB LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of jobs.

```
GET /api/v1/jobs/  
HTTP 200 OK  
Content-Type: application/json  
Vary: Accept  
Allow: GET, POST, HEAD, OPTIONS  
X-API-Time: 0.052s
```

The resulting data structure contains:

```
{  
  "count": 99,  
  "next": null,  
  "previous": null,  
  "results": [  
    ...  
  ]  
}
```

The `count` field indicates the total number of jobs found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more job records.

18.1 Results

Each job data structure includes the following fields:

- `id`: Database ID for this job. (integer)
- **type: Data type for this job. (multiple choice)**
 - `job`: Playbook Run
- `url`: URL for this job. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `created`: Timestamp when this job was created. (datetime)
- `modified`: Timestamp when this job was last modified. (datetime)
- `name`: (string)

- description: (string)
- unified_job_template: (field)
- **launch_type: (multiple choice)**
 - manual: Manual
 - callback: Callback
 - scheduled: Scheduled
 - dependency: Dependency
- **status: (multiple choice)**
 - new: New
 - pending: Pending
 - waiting: Waiting
 - running: Running
 - successful: Successful
 - failed: Failed
 - error: Error
 - canceled: Canceled
- failed: (boolean)
- started: (datetime)
- finished: (datetime)
- elapsed: (decimal)
- job_explanation: (string)
- **job_type: (multiple choice)**
 - run: Run
 - check: Check
 - scan: Scan
- inventory: (field)
- project: (field)
- playbook: (string)
- credential: (field)
- cloud_credential: (field)
- forks: (integer)
- limit: (string)
- **verbosity: (multiple choice)**
 - 0: 0 (Normal)
 - 1: 1 (Verbose)
 - 2: 2 (More Verbose)

- 3: 3 (Debug)
- 4: 4 (Connection Debug)
- 5: 5 (WinRM Debug)
- extra_vars: (string)
- job_tags: (string)
- force_handlers: (boolean)
- skip_tags: (string)
- start_at_task: (string)
- job_template: (field)
- passwords_needed_to_start: (field)
- ask_variables_on_launch: (field)

18.2 Create Jobs

Make a POST request to this resource with the following job fields to create a new job:

- name: (string, required)
- description: (string, default="")
- **job_type: (multiple choice, required)**
 - run: Run (default)
 - check: Check
 - scan: Scan
- inventory: (field, default=None)
- project: (field, default=None)
- playbook: (string, default="")
- credential: (field, default=None)
- cloud_credential: (field, default=None)
- forks: (integer, default=0)
- limit: (string, default="")
- **verbosity: (multiple choice)**
 - 0: 0 (Normal) (default)
 - 1: 1 (Verbose)
 - 2: 2 (More Verbose)
 - 3: 3 (Debug)
 - 4: 4 (Connection Debug)
 - 5: 5 (WinRM Debug)
- extra_vars: (string, default="")

- `job_tags`: (string, default="")
- `force_handlers`: (boolean, default=False)
- `skip_tags`: (string, default="")
- `start_at_task`: (string, default="")
- `job_template`: (field, default=None)

If the `job_template` field is specified, any fields not explicitly provided for the new job (except name and description) will use the default values from the job template.

18.3 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

AD HOC COMMAND LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of ad hoc commands.

```
GET /api/v1/ad_hoc_commands/  
HTTP 200 OK  
Content-Type: application/json  
Vary: Accept  
Allow: GET, POST, HEAD, OPTIONS  
X-API-Time: 0.053s
```

The resulting data structure contains:

```
{  
  "count": 99,  
  "next": null,  
  "previous": null,  
  "results": [  
    ...  
  ]  
}
```

The `count` field indicates the total number of ad hoc commands found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more ad hoc command records.

19.1 Results

Each ad hoc command data structure includes the following fields:

- `id`: Database ID for this ad hoc command. (integer)
- **type**: **Data type for this ad hoc command. (multiple choice)** -`ad_hoc_command`: Command
- `url`: URL for this ad hoc command. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `created`: Timestamp when this ad hoc command was created. (datetime)
- `modified`: Timestamp when this ad hoc command was last modified. (datetime)
- `name`: (string)
- **launch_type**: (multiple choice)

- manual: Manual
- callback: Callback
- scheduled: Scheduled
- dependency: Dependency
- **status: (multiple choice)**
 - new: New
 - pending: Pending
 - waiting: Waiting
 - running: Running
 - successful: Successful
 - failed: Failed
 - error: Error
 - canceled: Canceled
- failed: (boolean)
- started: (datetime)
- finished: (datetime)
- elapsed: (decimal)
- job_explanation: (string)
- **job_type: (multiple choice)**
 - run: Run
 - check: Check
 - scan: Scan
- inventory: (field)
- limit: (string)
- credential: (field)
- **module_name: (multiple choice)**
 - command
 - shell
 - yum
 - apt
 - apt_key
 - apt_repository
 - apt_rpm
 - service
 - group
 - user

- mount
- ping
- selinux
- setup
- win_ping
- win_service
- win_updates
- win_group
- win_user
- `module_args`: (string)
- `forks`: (integer)
- **verbosity**: (multiple choice)
 - 0: 0 (Normal)
 - 1: 1 (Verbose)
 - 2: 2 (More Verbose)
 - 3: 3 (Debug)
 - 4: 4 (Connection Debug)
 - 5: 5 (WinRM Debug)
- `become_enabled`: (boolean)

19.2 Create Ad Hoc Commands

Make a POST request to this resource with the following ad hoc command fields to create a new ad hoc command:

- **job_type**: (multiple choice, required)
 - run: Run (default)
 - check: Check
 - scan: Scan
- `inventory`: (field, default=None)
- `limit`: (string, default="")
- `credential`: (field, default=None)
- **module_name**: (multiple choice)
 - command (default)
 - shell
 - yum
 - apt
 - apt_key

- apt_repository
- apt_rpm
- service
- group
- user
- mount
- ping
- selinux
- setup
- win_ping
- win_service
- win_updates
- win_group
- win_user
- `module_args`: (string, default="")
- `forks`: (integer, default=0)
- **verbosity**: (multiple choice)
 - 0: 0 (Normal) (default)
 - 1: 1 (Verbose)
 - 2: 2 (More Verbose)
 - 3: 3 (Debug)
 - 4: 4 (Connection Debug)
 - 5: 5 (WinRM Debug)
- `become_enabled`: (boolean, default=False)

(Ad hoc commands were added in Ansible Tower version 2.2.0).

19.3 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

SYSTEM JOB TEMPLATE LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of system job templates.

```
GET /api/v1/system_job_templates/  
HTTP 200 OK  
Content-Type: application/json  
Vary: Accept  
Allow: GET, HEAD, OPTIONS  
X-API-Time: 0.060s
```

The resulting data structure contains:

```
{  
  "count": 99,  
  "next": null,  
  "previous": null,  
  "results": [  
    ...  
  ]  
}
```

The count field indicates the total number of system job templates found for the given query. The next and previous fields provides links to additional results if there are more than will fit on a single page. The results list contains zero or more system job template records.

20.1 Results

Each system job template data structure includes the following fields:

- id: Database ID for this system job template. (integer)
- **type: Data type for this system job template. (multiple choice)**
 - system_job_template: System Job Template
- url: URL for this system job template. (string)
- related: Data structure with URLs of related resources. (object)
- summary_fields: Data structure with name/description for related resources. (object)
- created: Timestamp when this system job template was created. (datetime)
- modified: Timestamp when this system job template was last modified. (datetime)
- name: (string)

- `description`: (string)
- `last_job_run`: (datetime)
- `last_job_failed`: (boolean)
- `has_schedules`: (boolean)
- `next_job_run`: (datetime)
- **status**: (multiple choice)
 - `new`: New
 - `pending`: Pending
 - `waiting`: Waiting
 - `running`: Running
 - `successful`: Successful
 - `failed`: Failed
 - `error`: Error
 - `canceled`: Canceled
 - `never updated`: Never Updated
 - `ok`: OK
 - `missing`: Missing
 - `none`: No External Source
 - `updating`: Updating
- **job_type**: (multiple choice)
 - `"`: _____
 - `cleanup_jobs`: Remove jobs older than a certain number of days
 - `cleanup_activitystream`: Remove activity stream entries older than a certain number of days
 - `cleanup_deleted`: Purge previously deleted items from the database
 - `cleanup_facts`: Purge and/or reduce the granularity of system tracking data

20.2 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

SYSTEM JOB LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of system jobs.

```
GET /api/v1/system_jobs/
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, POST, HEAD, OPTIONS
X-API-Time: 0.049s
```

The resulting data structure contains:

```
{
  "count": 99,
  "next": null,
  "previous": null,
  "results": [
    ...
  ]
}
```

The `count` field indicates the total number of system jobs found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more system job records.

21.1 Results

Each system job data structure includes the following fields:

- `id`: Database ID for this system job. (integer)
- **type: Data type for this system job. (multiple choice)**
 - `system_job`: Management Job
- `url`: URL for this system job. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `created`: Timestamp when this system job was created. (datetime)
- `modified`: Timestamp when this system job was last modified. (datetime)
- `name`: (string)

- description: (string)
- unified_job_template: (field)
- **launch_type: (multiple choice)**
 - manual: Manual
 - callback: Callback
 - scheduled: Scheduled
 - dependency: Dependency
- **status: (multiple choice)**
 - new: New
 - pending: Pending
 - waiting: Waiting
 - running: Running
 - successful: Successful
 - failed: Failed
 - error: Error
 - canceled: Canceled
- failed: (boolean)
- started: (datetime)
- finished: (datetime)
- elapsed: (decimal)
- job_explanation: (string)
- system_job_template: (field)
- **job_type: (multiple choice)**
 - " ": _____
 - cleanup_jobs: Remove jobs older than a certain number of days
 - cleanup_activitystream: Remove activity stream entries older than a certain number of days
 - cleanup_deleted: Purge previously deleted items from the database
 - cleanup_facts: Purge and/or reduce the granularity of system tracking data
- extra_vars: (string)

21.2 Create System Jobs

Make a POST request to this resource with the following system job fields to create a new system job:

- name: (string, required)
- description: (string, default="")
- system_job_template: (field, default=None)

- **job_type:** (multiple choice)
 - "": _____ (default)
 - cleanup_jobs: Remove jobs older than a certain number of days
 - cleanup_activitystream: Remove activity stream entries older than a certain number of days
 - cleanup_deleted: Purge previously deleted items from the database
 - cleanup_facts: Purge and/or reduce the granularity of system tracking data
- extra_vars: (string, default="")

21.3 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

SCHEDULES API ENDPOINT

Make a GET request to this resource to retrieve the list of schedules.

```
GET /api/v1/schedules/
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, HEAD, OPTIONS
X-API-Time: 0.050s
```

The resulting data structure contains:

```
{
  "count": 99,
  "next": null,
  "previous": null,
  "results": [
    ...
  ]
}
```

The `count` field indicates the total number of schedules found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more schedule records.

22.1 Results

Each schedule data structure includes the following fields:

- `id`: Database ID for this schedule. (integer)
- **type: Data type for this schedule. (multiple choice)**
 - `schedule`: Schedule
- `url`: URL for this schedule. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `created`: Timestamp when this schedule was created. (datetime)
- `modified`: Timestamp when this schedule was last modified. (datetime)
- `name`: (string)

- `description`: (string)
- `unified_job_template`: (field)
- `enabled`: (boolean)
- `dtstart`: (datetime)
- `dtend`: (datetime)
- `rrule`: (string)
- `next_run`: (datetime)
- `extra_data`: (field)

22.2 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

UNIFIED JOB TEMPLATE LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of unified job templates.

```
GET /api/v1/unified_job_templates/  
HTTP 200 OK  
Content-Type: application/json  
Vary: Accept  
Allow: GET, HEAD, OPTIONS  
X-API-Time: 0.055s
```

The resulting data structure contains:

```
{  
  "count": 99,  
  "next": null,  
  "previous": null,  
  "results": [  
    ...  
  ]  
}
```

The count field indicates the total number of unified job templates found for the given query. The next and previous fields provides links to additional results if there are more than will fit on a single page. The results list contains zero or more unified job template records.

23.1 Results

Each unified job template data structure includes the following fields:

- id: Database ID for this unified job template. (integer)
- **type: Data type for this unified job template. (multiple choice)**
 - project: Project
 - inventory_source: Inventory Source
 - job_template: Job Template
 - system_job_template: System Job Template
- url: URL for this unified job template. (string)
- related: Data structure with URLs of related resources. (object)
- summary_fields: Data structure with name/description for related resources. (object)

- `created`: Timestamp when this unified job template was created. (datetime)
- `modified`: Timestamp when this unified job template was last modified. (datetime)
- `name`: (string)
- `description`: (string)
- `last_job_run`: (datetime)
- `last_job_failed`: (boolean)
- `has_schedules`: (boolean)
- `next_job_run`: (datetime)
- **status**: (multiple choice)
 - `new`: New
 - `pending`: Pending
 - `waiting`: Waiting
 - `running`: Running
 - `successful`: Successful
 - `failed`: Failed
 - `error`: Error
 - `canceled`: Canceled
 - `never updated`: Never Updated
 - `ok`: OK
 - `missing`: Missing
 - `none`: No External Source
 - `updating`: Updating

23.2 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

UNIFIED JOB LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of unified jobs.

```
GET /api/v1/unified_jobs/  
HTTP 200 OK  
Content-Type: application/json  
Vary: Accept  
Allow: GET, HEAD, OPTIONS  
X-API-Time: 0.063s
```

The resulting data structure contains:

```
{  
  "count": 99,  
  "next": null,  
  "previous": null,  
  "results": [  
    ...  
  ]  
}
```

The `count` field indicates the total number of unified jobs found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more unified job records.

Note: The rules of encryption and decryption for Ansible Tower also apply to one field outside of credentials, the Unified Job `start_args` field, which is used through the `job`, `ad_hoc_command`, and `system_job` data types. Refer to [Understanding How Credentials Work](#) in the *Ansible Tower User Guide* for more information.

24.1 Results

Each unified job data structure includes the following fields:

- `id`: Database ID for this unified job. (integer)
- **type: Data type for this unified job. (multiple choice)**
 - `project_update`: SCM Update
 - `inventory_update`: Inventory Sync
 - `job`: Playbook Run

- ad_hoc_command: Command
 - system_job: Management Job
- url: URL for this unified job. (string)
- related: Data structure with URLs of related resources. (object)
- summary_fields: Data structure with name/description for related resources. (object)
- created: Timestamp when this unified job was created. (datetime)
- modified: Timestamp when this unified job was last modified. (datetime)
- name: (string)
- description: (string)
- unified_job_template: (field)
- **launch_type: (multiple choice)**
 - manual: Manual
 - callback: Callback
 - scheduled: Scheduled
 - dependency: Dependency
- **status: (multiple choice)**
 - new: New
 - pending: Pending
 - waiting: Waiting
 - running: Running
 - successful: Successful
 - failed: Failed
 - error: Error
 - canceled: Canceled
- failed: (boolean)
- started: (datetime)
- finished: (datetime)
- elapsed: (decimal)
- job_explanation: (string)

24.2 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

ACTIVITY STREAM LIST API ENDPOINT

Make a GET request to this resource to retrieve the list of activity streams.

```
GET /api/v1/activity_stream/  
HTTP 402 PAYMENT REQUIRED  
Content-Type: application/json  
Vary: Accept  
Allow: GET, HEAD, OPTIONS  
X-API-Time: 0.044s
```

The resulting data structure contains:

```
{  
  "count": 99,  
  "next": null,  
  "previous": null,  
  "results": [  
    ...  
  ]  
}
```

The `count` field indicates the total number of activity streams found for the given query. The `next` and `previous` fields provides links to additional results if there are more than will fit on a single page. The `results` list contains zero or more activity stream records.

25.1 Results

Each activity stream data structure includes the following fields:

- `id`: Database ID for this activity stream. (integer)
- **type: Data type for this activity stream. (multiple choice)**
 - `activity_stream`: Activity Stream
- `url`: URL for this activity stream. (string)
- `related`: Data structure with URLs of related resources. (object)
- `summary_fields`: Data structure with name/description for related resources. (object)
- `timestamp`: (datetime)
- **operation: The action taken with respect to the given object(s). (multiple choice)**
 - `create`: Entity Created

- update: Entity Updated
- delete: Entity Deleted
- associate: Entity Associated with another Entity
- disassociate: Entity was Disassociated with another Entity
- changes: A summary of the new and changed values when an object is created, updated, or deleted (field)
- object1: For create, update, and delete events this is the object type that was affected. For associate and disassociate events this is the object type associated or disassociated with object2 (string)
- object2: Unpopulated for create, update, and delete events. For associate and disassociate events this is the object type that object1 is being associated with (string)

25.2 Reviewing the Options Endpoint

The *Options Endpoint* table offers a view of the Options for this endpoint. When viewing the endpoint in the browsable API, clicking the “Options” button gives you the raw JSON for the following:

INDEX

- genindex

COPYRIGHT © 2016 RED HAT, INC.

Ansible, Ansible Tower, Red Hat, and Red Hat Enterprise Linux are trademarks of Red Hat, Inc., registered in the United States and other countries.

If you distribute this document, or a modified version of it, you must provide attribution to Red Hat, Inc. and provide a link to the original version.

Third Party Rights

Ubuntu and Canonical are registered trademarks of Canonical Ltd.

The CentOS Project is copyright protected. The CentOS Marks are trademarks of Red Hat, Inc. (“Red Hat”).

Microsoft, Windows, Windows Azure, and Internet Explore are trademarks of Microsoft, Inc.

VMware is a registered trademark or trademark of VMware, Inc.

Rackspace trademarks, service marks, logos and domain names are either common-law trademarks/service marks or registered trademarks/service marks of Rackspace US, Inc., or its subsidiaries, and are protected by trademark and other laws in the United States and other countries.

Amazon Web Services”, “AWS”, “Amazon EC2”, and “EC2”, are trademarks of Amazon Web Services, Inc. or its affiliates.

OpenStack™ and OpenStack logo are trademarks of OpenStack, LLC.

Chrome™ and Google Compute Engine™ service registered trademarks of Google Inc.

Safari® is a registered trademark of Apple, Inc.

Firefox® is a registered trademark of the Mozilla Foundation.

All other trademarks are the property of their respective owners.

A

- activity stream
 - API endpoint, 68
 - options endpoint, 69
 - results, 68
- ad hoc commands
 - API endpoint, 53
 - create, 55
 - options endpoint, 56
 - results, 53
- API
 - browsable, 2
 - JSON, 5
 - POST, 5
 - PUT, 5
 - root directory, 5
- API endpoint
 - activity stream, 68
 - ad hoc commands, 53
 - auth token, 12
 - configuration, 15
 - credential list, 31
 - current user (me), 17
 - dashboard, 19
 - group list, 41
 - host list, 43
 - inventory list, 34
 - inventory script list, 36
 - inventory source list, 38
 - job list, 49
 - job template list, 45
 - me (current user), 17
 - organizations list, 20
 - ping, 14
 - project list, 26
 - schedules, 62
 - system job list, 59
 - system job template list, 57
 - team list, 29
 - unified job list, 66
 - unified job template list, 64
 - user list, 23

- auth token
 - API endpoint, 12
 - options endpoint, 13

B

- browsable API, 2

C

- configuration
 - API endpoint, 15
 - options endpoint, 16
- content type
 - JSON, 5
- conventions, 5
- create
 - ad hoc commands, 55
 - credential list, 32
 - group list, 42
 - host list, 44
 - inventory list, 35
 - inventory script list, 37
 - job list, 51
 - job template list, 47
 - organizations list, 21
 - project list, 28
 - system job list, 60
 - team list, 30
 - user list, 24
- credential list
 - API endpoint, 31
 - create, 32
 - options endpoint, 33
 - results, 31
- current user (me)
 - API endpoint, 17
 - options endpoint, 18

D

- dashboard
 - API endpoint, 19
 - options endpoint, 19

F

filtering, 6

G

group list

- API endpoint, 41
- create, 42
- options endpoint, 42
- results, 41

H

host list

- API endpoint, 43
- create, 44
- options endpoint, 44
- results, 43

I

inventory list

- API endpoint, 34
- create, 35
- options endpoint, 35
- results, 34

inventory script list

- API endpoint, 36
- create, 37
- options endpoint, 37
- results, 36

inventory source list

- API endpoint, 38
- options endpoint, 40
- results, 38

J

job list

- API endpoint, 49
- create, 51
- options endpoint, 52
- results, 49

job template list

- API endpoint, 45
- create, 47
- options endpoint, 48
- results, 45

JSON

- API, 5
- content type, 5

M

me (current user)

- API endpoint, 17
- options endpoint, 18

O

options endpoint

- activity stream, 69
- ad hoc commands, 56
- auth token, 13
- configuration, 16
- credential list, 33
- current user (me), 18
- dashboard, 19
- group list, 42
- host list, 44
- inventory list, 35
- inventory script list, 37
- inventory source list, 40
- job list, 52
- job template list, 48
- me (current user), 18
- organizations list, 22
- ping, 14
- project list, 28
- schedules, 63
- system job list, 61
- system job template list, 58
- team list, 30
- unified job list, 67
- unified job template list, 65
- user list, 25

ordering

- sorting, 6

organizations list

- API endpoint, 20
- create, 21
- options endpoint, 22
- results, 21

P

pagination, 8

ping

- API endpoint, 14
- options endpoint, 14

POST

- API, 5

project list

- API endpoint, 26
- create, 28
- options endpoint, 28
- results, 26

PUT

- API, 5

Q

queryset, 6

R

- read-only fields, 9
- results
 - activity stream, 68
 - ad hoc commands, 53
 - credential list, 31
 - group list, 41
 - host list, 43
 - inventory list, 34
 - inventory script list, 36
 - inventory source list, 38
 - job list, 49
 - job template list, 45
 - organizations list, 21
 - project list, 26
 - schedules, 62
 - system job template list, 57
 - team list, 29
 - unified job list, 66
 - unified job template list, 64
 - user list, 24
- root directory
 - API, 5

S

- schedules
 - API endpoint, 62
 - options endpoint, 63
 - results, 62
- searching, 6
- serializer, 8
- sorting
 - ordering, 6
- system job list
 - API endpoint, 59
 - create, 60
 - options endpoint, 61
- system job template list
 - API endpoint, 57
 - options endpoint, 58
 - results, 57

T

- team list
 - API endpoint, 29
 - create, 30
 - options endpoint, 30
 - results, 29
- tools, 2
 - tower-cli, 9
- tower-cli, 9
 - capabilities, 9
 - installation, 9

U

- unified job list
 - API endpoint, 66
 - options endpoint, 67
 - results, 66
- unified job template list
 - API endpoint, 64
 - options endpoint, 65
 - results, 64
- user list
 - API endpoint, 23
 - create, 24
 - options endpoint, 25
 - results, 24