# 2. Query the artifacts

## 2.1. Basic usage

We have an access key to initiate authenticated calls to the API (if it is not the case, please go back to Authentication), we can now query a tracker to retrieve some artifacts.

For example to retrieve all artifacts that are in *Completed* status we will issue the following query:

```
$ curl -XGET --header 'Content-type: application/json' \
    --header 'X-Auth-AccessKey: tlp.k1.1026…' \
    -d '{"status_id":{"operator":"contains","value":"504"}}' \
    https://tuleap.example.com/api/trackers/101/artifacts
```

This will return a bunch of artifacts. Some explanations about values used in this example:

- `tlp.k1.1026…` and `115` is the access key you get in the previous section. You should already know what it is and how to get it.
- `status_id` is the shortname of the field *Status* of my tracker.
- `504` is the id of the value *Completed*. This value is one of the possible values for the field *Status*.
- `101` in the URI is the id of the tracker.

In the list of artifact representations we get, we can query the API to have more details about an artifact with `GET /api/artifacts/220`.

> Note
>
> The API is self discoverable. This means that each resource will detail its sub-resources, meaning that you shouldn't have to know in advance which id or which shortname to use in your calls.
>
> For example, `GET /api/projects` will return an array of project representations:
>
> ```
> [{
>   "id": 110,
>   "uri": "projects/110",
>   "label": "Project 42",
>   "shortname": "project42",
>   "resources": [
>     {
>       "type": "trackers",
>       "uri": "projects/110/trackers"
>     }
>   ]
> }]
> ```
>
> From here I can get the list of trackers of the project with `GET /api/projects/110/trackers`, etc. Which calls you have to orchestrate will depend on your business case.
>
> In our example, getting the structure of the tracker is important as we need to know the id of the fields/values (either `projects/110/trackers` or `trackers/123` will give us the structure).

## 2.2. Pagination

Most of the collections (collections of trackers, collections of artifacts, …) returned by the API are paginated. This means that only a bunch of results is returned (10, 50 or 100, it depends on the route). Information about the current pagination is returned in response headers. You can use `--include` for `curl` command to get the headers.

- The `LIMIT` is the limit used to return the items i.e. the chunk size. Here only 100 items are returned for each call. The limit cannot exceed `LIMIT-MAX`.
- The `OFFSET` is the offset used to return the items. The first call we will retrieve the items from 0 to 99 (offset = 0, limit = 100), the second call we will retrieve the items from 100 to 199 (offset = 100, limit = 100), etc.
- The `SIZE` is the total size of the collection. It is used to know how many calls we have to do to get all the collection.

Routes that return paginated collection accept two parameters that we will adjust to our needs: `limit` and `offset`:

```
$ curl -XGET --header 'Content-type: application/json' \
    --header 'X-Auth-AccessKey: tlp.k1.1026…' \
    -d '{"status_id":{"operator":"contains","value":"504"}}' \
    https://tuleap.example.com/api/trackers/101/artifacts?offset=100&limit=50
```

> Note
>
> Pagination is here to save resources (server, bandwidth, client) by not returning all items at once. You can adjust the limit to retrieve more items or less items at once. However do not ask too few items in order to not penalize user experience (YAY network latency).

## 2.3. Conclusions

Now we are able to:

- Do authenticated calls
- Do a basic GET to retrieve information (projects, trackers, artifacts, …)
- Handle pagination

This only covered artifacts, but you can apply what you've just learned to every routes provided by your Tuleap instance.

You can stop here or do write operations in next section: Update an artifact.

Previous Next

enalean software engineering is now
Tuleap project and this website are sponsored by Enalean