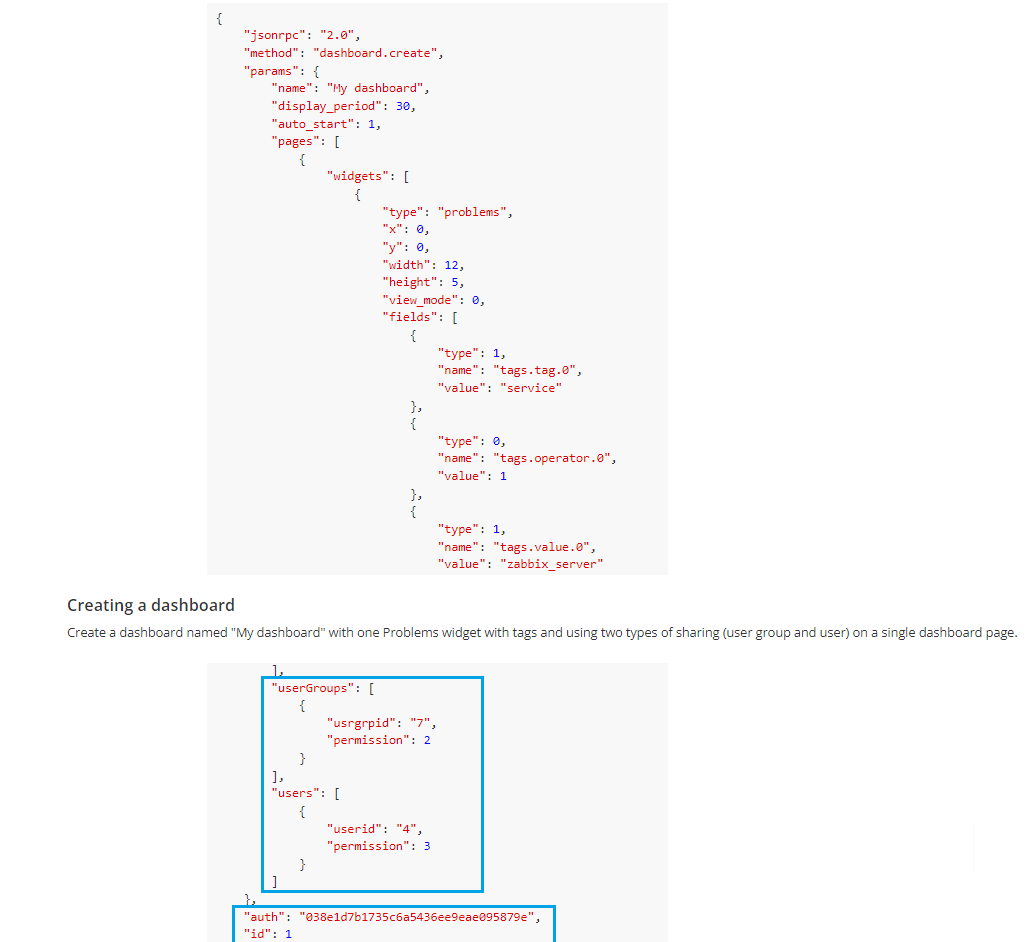
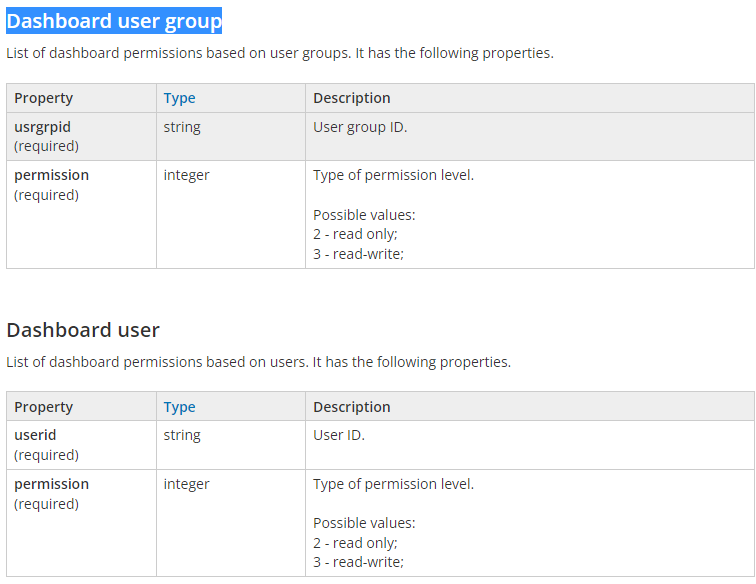
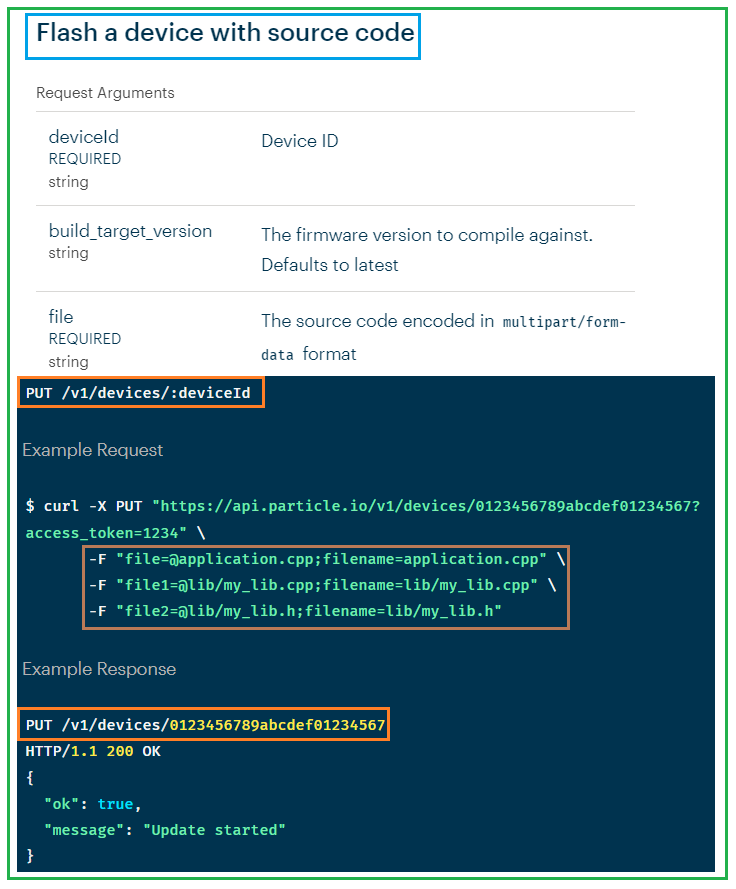
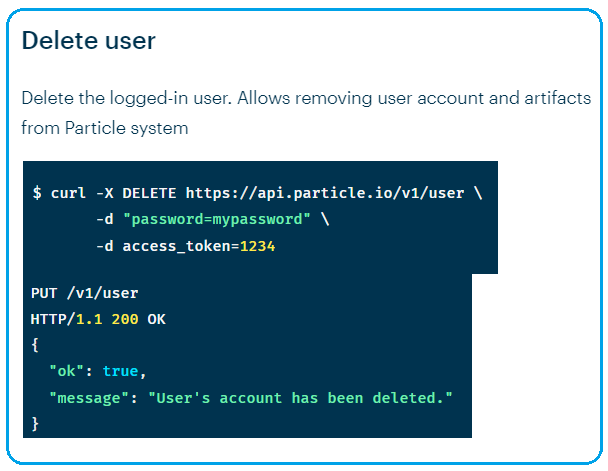
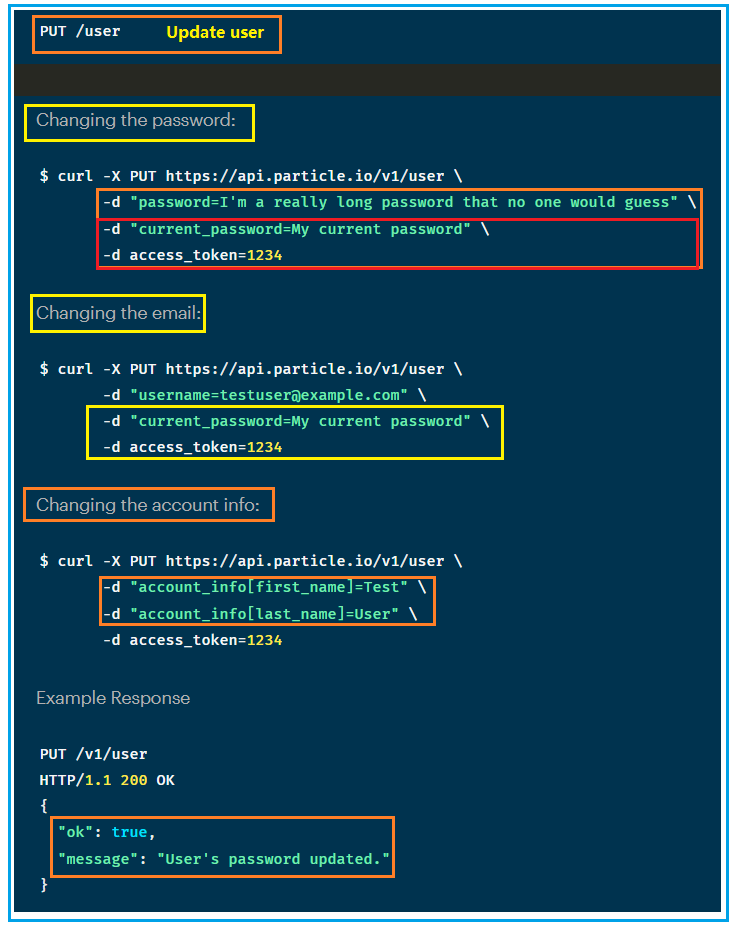
8/6/2022, 8/11, 8/13, 8/15,

1. Projects
2. Test tools
3. Test management tools
4. Test automation pytest
5. Selenium
6. Figure interconnections between user/device/workflow/token/test/projects(team)
7. Projects
8. <https://www.zabbix.com/documentation/current/en/manual/api/reference/dashboard/create>
9. <https://docs.particle.io/reference/cloud-apis/api/#list-devices-in-a-product>







<https://www.zabbix.com/documentation/current/en/manual/web_interface/frontend_sections/administration>

1. <https://docs.particle.io/reference/cloud-apis/api/#user>
2. Test tools
3. Test management tools
4. <https://blog.testproject.io/2021/02/15/built-in-cloud-reporting-tool-for-pytest-and-unittest/>
5. <https://www.eswcompany.com/python-test-tools-should-you-be-using/>
6. <https://www.fugue.co/blog/blog-post-creating-an-automated-cloud-infrastructure-testing-tool-with-terraform-and-pytest>
7. Test automation pytest

# [How can I write Python code to support both Windows and Linux](https://stackoverflow.com/questions/58670743/how-can-i-write-python-code-to-support-both-windows-and-linux)

# <https://stackoverflow.com/questions/58670743/how-can-i-write-python-code-to-support-both-windows-and-linux>

# how to maintain one python code base in both linux and windows

1. Selenium
2. Projects
3. User

* User/team/
* During internal testing/customer feed back, noticed that user/credential with group to manage devices is not efficient when devices are shared
* Recommend add API team, which is project based.
* Team/projects, within a project or team, all users have same access rights to the devices, tests, and work\_flows.
* Project manager is created as superuser. Who can create team, add users to the team, add resources/ devices etc.
* ;;
* Within a group, superuser vs user
* Superuser can assign devices/test/test\_flows ownership to other user
* User can only transfer ownership to other user.
* When user creates device, work\_flow, report; they can set the access to read only or read and write. Superuser can overwrite the access property.

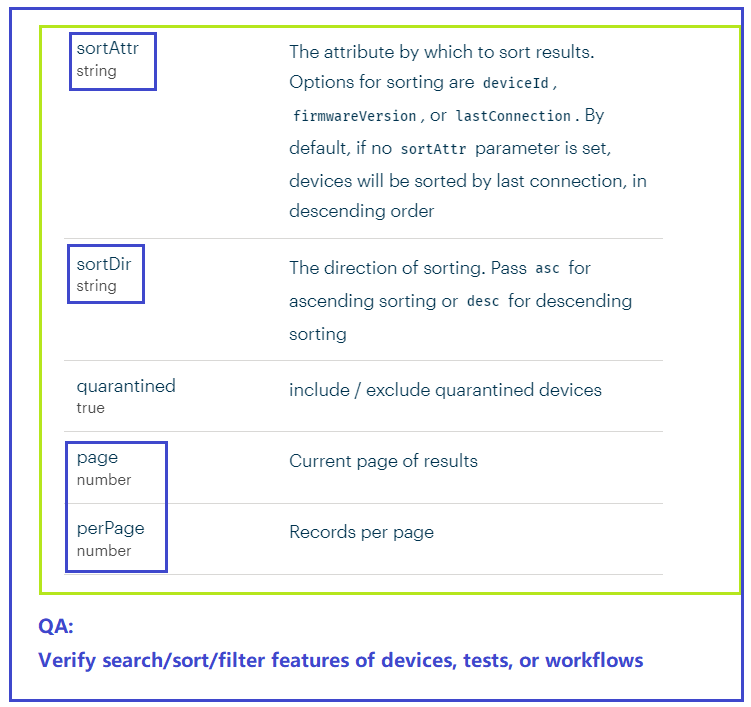
1. Devices

* List all devices that are part of a team(project) or user. Results are paginated, by default returns 25 device records per page
* Curl [https://server:port/v1/teams/{teamID}/devices?access\_token=1234](https://server:port/v1/teams/%7bteamID%7d/devices?access_token=1234)
* **Search/Filter/Sort**
* Show devices from a project and specific user

Curl https:/server:port/v1/devices?team=teamID&user=userID





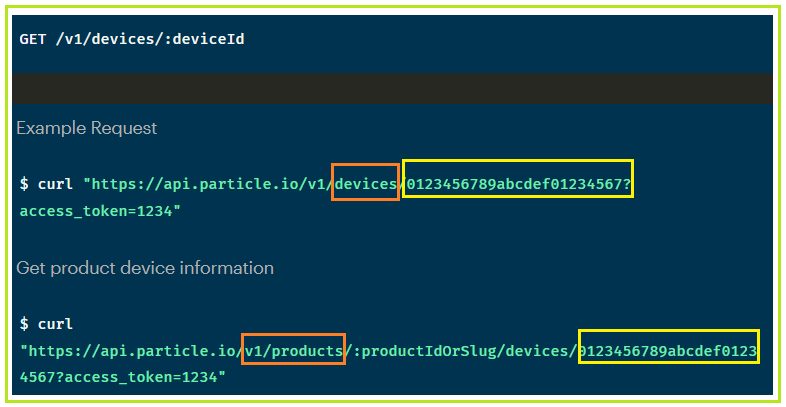


* Verify **inter-connect** between APIs: same data should be consistent via different API calls.

V1/team/devices: shows all the teams with their corresponding assigned devices

/api/v1/devices: show devices and which team they are assigned to if any

So if a device is removed/updated from devices, the changes should automatically reflected from its assigned teams.



--

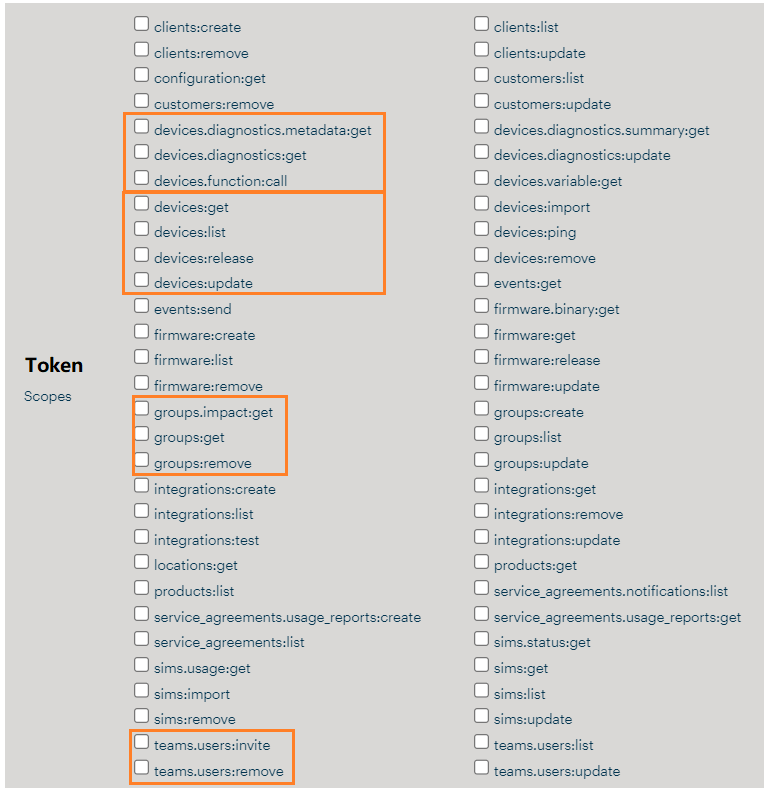
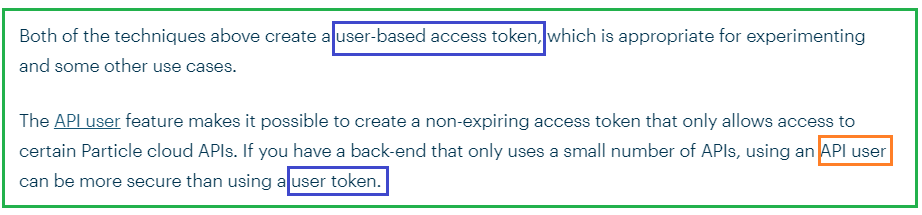
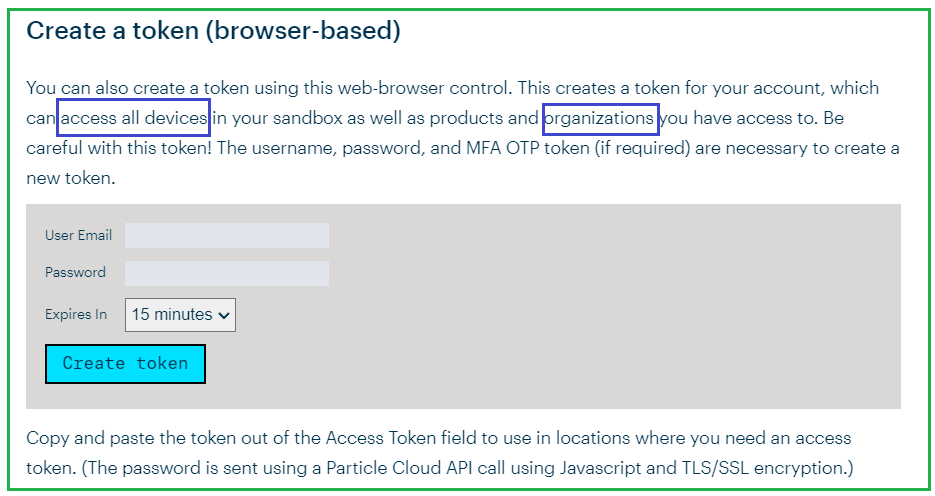


* **Cross-check API calls**

****

1. Token

* Token serves mainly for API access as an identity metric. It carries the same rights according to its user.
* Our token is user based, not api call based.
* <https://docs.particle.io/reference/cloud-apis/access-tokens/>



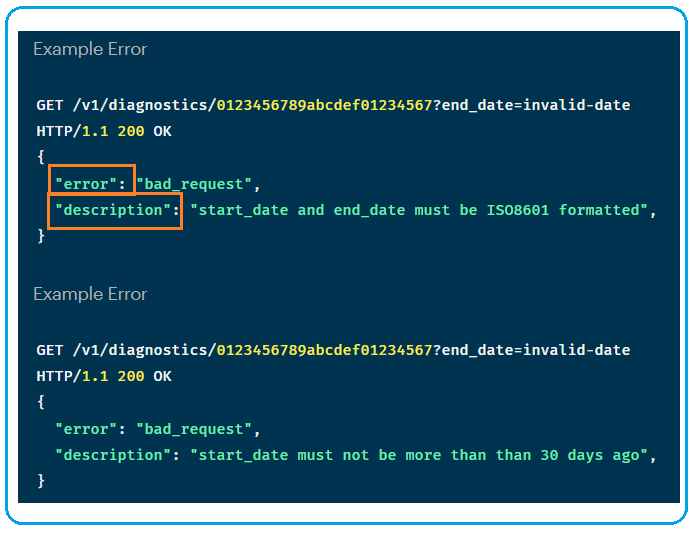
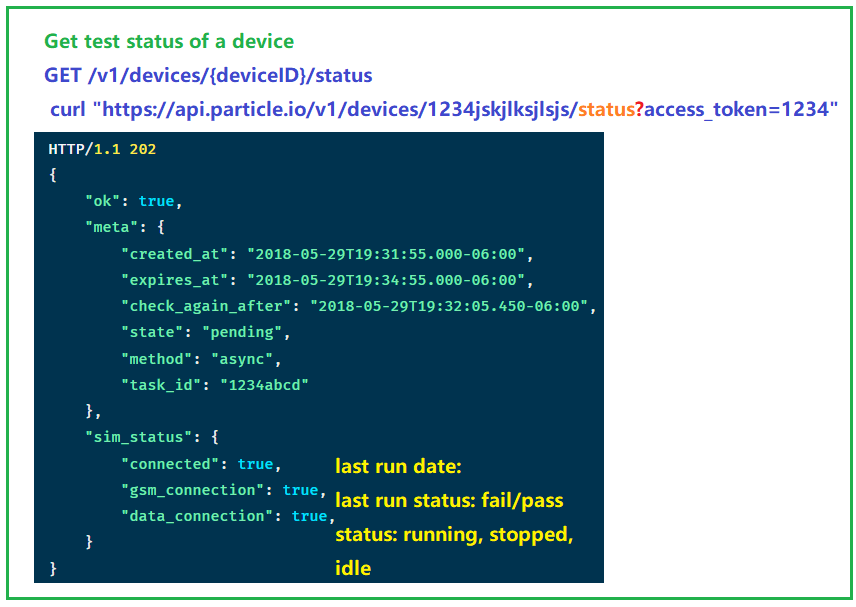
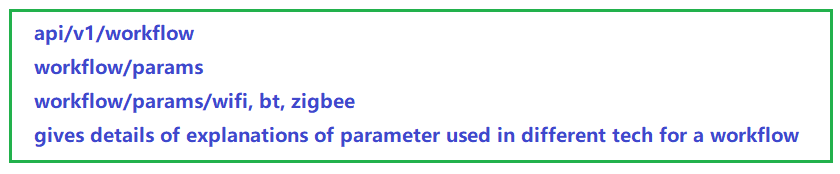
1. Error Checking of response

* A good source as resume material.

1. Api/test, api/work\_flow, api\_report

* **Get historical tests run by a device**



****

1. A