How-To Create Custom Quick Actions in opConfig

Step 1. Create your Command Set

Before creating the quick action you will need to have your [Command Set](#) file created. opConfig lets you organize whatever commands you'd like it to run into an arbitrary number of groups which we call command sets. Let's create a basic command set using df to check the disc space on a Linux device for the purposes of this How-To however, these command sets can be as simple or as complicated as needed to complete the desired tasks.

Command sets are stored in the single file `conf/command_sets.nmis` in opConfig before 2.2.4, and since 2.2.4 opConfig supports this *and* individual command set files in `conf/command_sets.d/`. It is recommended that you use individual files (as that makes things easier to maintain and more robust, e.g. should one file have syntactical problems). For backwards compatibility the old single file is consulted first, then any individual ones. opConfig loads a command set when it's seen for the **first** time, and any subsequent occurrences of a clashing config set (i.e. same name) will be ignored but logged.

Navigate to the /usr/local/omk/conf/command_sets.d directory and create a new Command Set file. For this example I will be creating the file df_test.nmis.

```
cd /usr/local/omk/conf/command_sets.d
vi df_test.nmis
```

A command set definition consists of these types of information:

- meta-information like the command set's name (which can be used in CRON to determine when / how often this command set is run),
- optional control information like scheduling constraints and an expiration policy for the commands in that set,
- filters for selecting which devices/nodes the command set should apply to. These are discussed in more detail below.
- and finally the list of the actual commands to be run (which may also have certain modifiers and meta-data attached to them).

Below is an example of the Command Set file created for this How-To. This Command Set is running a simple df for Linux devices to display the amount of available disk space. You may copy this code to follow along or edit it to suite your needs:

```
%hash = (

  'DF_Test' => {
    'os_info' => {
      'os' => '/(Linux|CentOS|Ubuntu)/'
    },
    'scheduling_info' => {
      'run_commands_on_separate_connection' => 'false',
    },
    'commands' => [
      {
        'privileged' => 'false',
        'command' => 'df',
        'exec' => 'df',
        'tags' => ['Linux','DF','operations'],
      },
    ],
  },
);
```

Defining which nodes a command set applies to

This is normally done by filtering by OS info, but not necessarily limited to OS info only:

```
#filter using both regular expression and string equality checks:
'os_info' => {
 'version' => '/12.2|12.4|15.0/',
 'os' => 'IOS'
},
# or just the specific os family/type and that's all:
'os_info' => {
 'os' => '/(Linux|CentOS|Ubuntu)/'
},
```

The examples above specify that the node in question must have an os_info property set, with sub-properties os and version in the first example and just os in the second.

Plain string values indicate string comparison, but regular expressions can be given too. A node is only considered for this particular command set if all filtering expressions match. You can filter on any node property, not just properties from os_info (but the default command sets only use os_info).

Prior to version 3.1.1, opConfig considered command sets without filter blocks as disabled; for these versions you may want a 'wildcard' filter matching anything whatsoever, which can be achieved by adding an `os_info` filter block with a 'match-everything' regular expression for `os`, e.g `'os' => '/.*/'`.

From opConfig 3.1.1 onwards a command set without filter is interpreted as to apply to all nodes without any restriction.

More information on Command Sets can be found HERE

Step 2: Create your Quick Action

Quick Actions

Quick actions are templates for new Virtual Operator jobs, we have shipped four sample jobs but you can create your own. Clicking the quick action button will take you too a new virtual operator screen and fill out the specified fields. The default four jobs are defined in the file: /usr/local/omk/lib/json/opConfig/table_schemas/opConfig_action-elements.json

Create your own Quick Action

You can create your own Quick Actions by copying the /usr/local/omk/lib/json/opConfig/table_schemas/opConfig_action-elements.json file to /usr/local/omk/conf/table_schemas/ (you may need to create the table_schemas folder if it does not exist) and then editing it. Available field options are described in the table below.

```
cp /usr/local/omk/lib/json/opConfig/table_schemas/opConfig_action-elements.json /usr/local/omk/conf
/table_schemas/
cd /usr/local/omk/conf/table_schemas
vi opConfig_action-elements.json
```
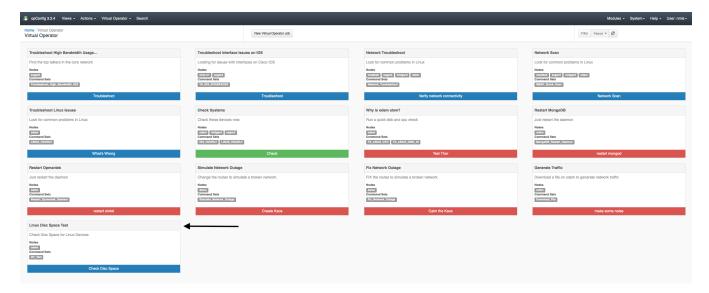
Once inside the opConfig_action-elements.json file, create a new section for the command set we created. Below is an example of the entry I added to the file:

```
{
"name": "Linux Disc Space Test",
"description": "Check Disc Space for Linux Devices",
"nodes": ["odem"],
"command_sets": ["DF_Test"],
"buttonLabel": "Check Disc Space",
"buttonClass": "btn-primary"
}
```
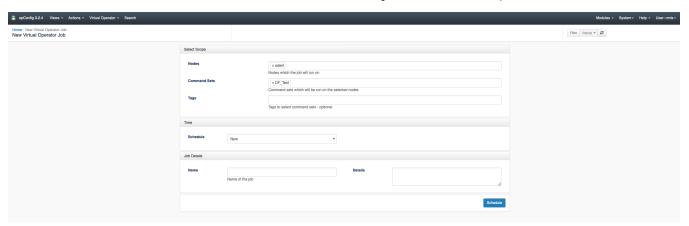
name	string	Name which is shown at the top of the quick action element
description	string	Text shown under the quick action name, useful to describe what the action does
command_sets	array of strings	Command set keys which you wish to be run
nodes	array of strings	Names of nodes which you wish the command sets to be run against
buttonLabel	string	Text of the run button
buttonClass	string	Css class applied to the button to colour it. • btn-default • btn-primary (default, blue) • btn-success (green) • btn-warning (orange) • btn-danger (red)

Step 3: Confirm your Quick Action is working

After adding your Quick Action entry to the opConfig_action-elements.json file, navigate to opConfig and select 'Virtual Operator'. As shown in the screenshot below the newly created Quick Action 'Linux Disc Space Test' is now displaying. Click on the button to start the Quick Action.

After clicking the Quick Action button the page below will auto fill the nodes provided in the opConfig_action-elements.json file as well as the Command Sets we created in the df_test.nmis file. Schedule a time for the Quick Action to run, give it a name and a description then click 'Schedule'.

Once the job has been completed you can view the results and details of the Quick Command. To view the command output click on the command in the Commands table as indicated by the arrow in the screenshot below.

The page below displays the command output as well as the Command Summary and the Most Recent Revisions. That's it, that's all there is to it! Your Quick Action is working!

Home / Job Result / Command Output
Command Output

Compare Revisions | Compare Command Outputs | Raw Output | Run Command Now

Filter 8d ▾ ⟳

Filter Command Outputs

Node	odem
Command	df
Revision	4181

Filter

⇄ Detect Changes   `Disabled`

≡ Command Summary

Job	DF_Test - odem
Revision	4181 `Unprotected`
Node	odem
Host	192.168.88.15
Command	df
Command Set	DF_Test
Created at	2020-04-27T19:32:43
Updated at	2020-04-27T19:32:43
Last Attempt at	2020-04-27T19:32:43

≡ odem O/S Summary

OS	Linux
Version	2.6.32-754.17.1.el6.x86_64
Major	2.6
Image	

≡ Most Recent Revisions

Date/Time	Revision
2020-04-27T19:32:43	4181
2020-04-27T19:02:46	4180
2020-04-27T19:00:10	4179
2020-04-27T18:19:43	4178
2020-04-27T18:02:45	4177

>_ Command Output

```
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/mapper/vg_nmis64-lv_root
                     16302888   5287652  10256376  35% /
tmpfs                 1470036         0   1470036   0% /dev/shm
/dev/sda1              487652    183468    278584  40% /boot
/dev/mapper/vg_nmis64_data-lv_data
                     41149760  15158016  23894000  39% /data
/dev/mapper/vg_nmis64-lv_var
                     20027260   7067668  11935592  38% /var
```