How to load objects into the Regional Controller

Two command line tools are provided to interface to the Regional Controller via the *bash* command line:

- *rc_cli* - which allows you to perform individual API calls
- *rc_loaddata* - used to do batch initialization of the RC database

Both tools require Python 3 be installed on your machine.

RC_CLI - Perform individual API calls

The options and arguments to the *rc_cli* command may be displayed by invoking it with *–help*:

```
$ ./rc_cli --help
usage: rc_cli [-h] [-H HOST] [-u USER] [-p PASSWORD] [-s SORT] [-F] [-Y]
              obj op [args [args ...]]

Perform API calls against a Regional Controller.

positional arguments:
  obj          The object to manipulate
  op           The operation to perform on the object
  args         Extra arguments for the operation

optional arguments:
  -h, --help   show this help message and exit
  -H HOST      the host of the regional controller (default localhost)
  -u USER      the username to use (default admin)
  -p PASSWORD  the password to use
  -s SORT      the field to sort list views by
  -F           forcibly delete a POD (use with "pod delete")
  -Y           display YAML columns in the output of list commands
```

rc_cli will first login to the RC using the host/user/password combination you specify, and then will perform the operation (specified by *op*) on the object (*obj*) listed. The user must have the appropriate permissions in the database to perform the requested operation. For example, in order to run the *pod create* command, the user must have a user role that includes the *create-pod* or *create-** role attributes.

Objects correspond to the objects that the RC controls; e.g. *blueprint*, *edgesite*, *hardware*, *node*, *pod*, and *region*. Operations are *create*, *delete*, *list*, and *sh ow*.

The *create* operation takes as arguments the names of one or more YAML files containing the objects to be created. For successfully created objects, it will print the URL of the newly created object, which contains the object's UUID. The YAML files provided must contain only YAML descriptions of the appropriate type specified on the command line. The format of these YAML objects can be determined by referencing Regional Controller API Documentation.

```
$ ./rc_cli -u admin -p admin123 hardware create hardware.yaml
https://localhost/api/v1/hardware/36a72db1-f2d9-424b-b94e-b72ca7a50cfe
```

delete takes as arguments a list of UUIDs of the objects to be deleted. If the specified objects are deleted, it produces no output.

list takes no arguments and just lists all objects of the specified type in the RC. The returned list will be sorted based on UUID, unless a different sort column is passed (via the *-s sort* option). The *list* operation will not display the YAML column of any object that has a YAML column unless the -Y option is provided.

```
$ ./rc_cli -u admin -p admin123 region list
UUID                                 Name       Description
Parent
------------------------------------  ---------  ------------------------------------------------------------
------------------------------------
00000000-0000-0000-0000-000000000000  Universal  The Parent of all Regions
00000000-0000-0000-0000-000000000000
3e2c604a-6ee7-4694-bbdc-9d19b146c6a0  Lab        The main region that all nodes in the lab will all be in.
00000000-0000-0000-0000-000000000000
efa82a42-c3b0-49c7-9db3-bbcd85927bbf  OEzone     A special region for all OpenEdge hardware in the lab.
00000000-0000-0000-0000-000000000000
```

show takes as arguments a list of UUIDs of the objects to show the details of.

```
$ ./rc_cli -u admin -p admin123 hardware show c1dfa1ac-53e0-11e9-86c2-c313482f1fdb
---
description: Standard Dell configuration for Rover/Unicycle
name: Dell PowerEdge R740
uuid: c1dfa1ac-53e0-11e9-86c2-c313482f1fdb
yaml:
  cpu: 2x22 Cores @ 2.1GHz Skylake 6152 CPU
  disk:
  - 4x480G SSD
  - 6x2.4T HDD
  lom: 4x10G Intel 710
  nic:
  - 2x1G LOM Intel 5xx
  - 2x25G PCI3 Intel 710
  ps: 2
  ram: 12x32GB
```

RC_LOADDATA - Batch Load Data into the RC DB

The options and arguments to the *rc_loaddata* command may be displayed by invoking it with *–help*:

```
$ ./rc_loaddata --help
usage: rc_loaddata [-h] [-H HOST] [-u USER] [-p PASSWORD]
                   [-s {hardware,regions,nodes,edgesites,blueprints,pods}]
                   [-A] [-D] [-M]
                   YAML [YAML ...]

Add/delete objects from a Regional Controller. The objects are retrieved from
one or more YAML files.

positional arguments:
  YAML                  YAML files containing RC objects to be
                        added/deleted/matched

optional arguments:
  -h, --help            show this help message and exit
  -H HOST               the host of the regional controller (default
                        localhost)
  -u USER               the username to use (default admin)
  -p PASSWORD           the password to use
  -s {hardware,regions,nodes,edgesites,blueprints,pods}
                        only process the named section of the YAMLs (default
                        all)
  -A                    add new entries in YAMLs to the RC
  -D                    delete entries in YAMLs from the RC
  -M                    match entries in YAMLs to the RC
```

The purpose of *rc_loaddata* is to perform operations on large number of RC objects at once. The operation to perform is specified by the -A/-D/-M options, only one of which may be provided.

- -A - add all entries in the YAML files to the RC database.
- -D - delete all entries in the YAML files from the RC database.
- -M - match entries between the YAML files and the RC database. That is, make the RC database identical to the contents of the catenated YAML files.

You can restrict the objects that the above operations are performed against by specifying one or more *-s section* options. Normally all sections are scanned.

When adding objects to the RC, the objects are added in a “bottom up” order; that is: ‘hardware’, ‘regions’, ‘nodes’, ‘edgesites’, ‘blueprints’, and finally ‘pods’. When deleting objects, the reverse order (top down) is used.

As with *rc_cli*, *rc_loaddata* will first login to the RC using the host/user/password combination you specify.

The YAML files themselves consists of stanzas for any of the six sections needed, followed by named objects describing the objects of each type to be added/deleted/matched. For example:

```
hardware:
  Dell_740xd:
    uuid: 9897a008-71d4-11e9-8bda-0017f20dbff8
    description: Dell 740xd hardware for the REC Blueprint
    yaml:
      todo: put hardware details here
      rack_layout:
        height: 2U

regions:
  zone1:
    description: The main region that all nodes in the MT lab will all be in.

edgesites:
  REC_OpenEdge1:
    uuid: 60ab1298-7769-11e9-92b3-373d9b2f2476
    description: The first OE REC cluster
    nodes: [ aknode201, aknode202, aknode203, aknode204, aknode205 ]
    regions: [ zone1 ]

blueprints:

nodes:
  aknode201:
    hardware: Nokia_OE19
    yaml:
      oob_ip: 172.26.16.201
      rack_location: { name: Row4_Rack6, slot: 2, unit: 1 }
```