New quantum client V2.0

Summary

Quantum client v2.0 adopts cliff framework, has flexibility to support V1.x and V2.0 Quantum API.It supports different set of commands according to given API version. The new quantum client's binary is quantumv2. This poster describes the commands for Quantum V2.0 API.

Features from cliff

Interactive mode

If there is no command is specified, the quantumv2 will enter into interactive mode:

```
# quantumv2 --os-username admin --os-password password --os-tenant-name admin
--os-auth-url http://localhost:5000/v2.0
(quantumv2) help

Shell commands (type help <topic>):
===================================
cmdenvironment  edit  hi      l   list  pause  r    save  shell     show
ed              help  history  li  load  py     run  set   shortcuts

Undocumented commands:
======================
EOF  eof  exit  q  quit

Application commands (type help <topic>):
=========================================
update_net   help         update_subnet  delete_subnet  list_ports   show_net
show_subnet  delete_port  list_nets      show_port      update_port
create_net   delete_net   create_subnet  list_subnets   create_port

(quantumv2) list_nets -F id -F name
+--------------------------------------+---------+
| id                                   | name    |
+--------------------------------------+---------+
| 358c43b7-2042-4f5e-848c-b88f441fc71a | test2   |
| 5cba94c1-8ed3-44ca-9f8c-a94fd9fc5c6a | test1_0 |
+--------------------------------------+---------+
```

Output format

We can use -h after each command to show the usage of each command:

```
(quantumv2) list_nets -h
usage: list_nets [-h] [-f {csv,html,json,table,yaml}] [-c COLUMN]
                 [--quote {all,minimal,none,nonnumeric}]
                 [--request-format {json,xml}] [-D] [-F FIELDS]
                 ...

List networks that belong to a given tenant Sample: list_nets -D --
--name=test4 --tag a b

positional arguments:
 filter_specs          filters options: --key1 [type=int|bool|...] value
                       [--key2 [type=int|bool|...] value ...]

optional arguments:
 -h, --help            show this help message and exit
 --request-format {json,xml}
                       the xml or json request format
 -D, --show-details    show detailed info
 -F FIELDS, --fields FIELDS
                       specify the field(s) to be returned by server, can be
                       repeated

output formatters:
 output formatter options

 -f {csv,html,json,table,yaml}, --format {csv,html,json,table,yaml}
                       the output format to use, defaults to table
 -c COLUMN, --column COLUMN
                       specify the column(s) to include, can be repeated

CSV Formatter:
 --quote {all,minimal,none,nonnumeric}
                       when to include quotes, defaults to nonnumeric
```

We can see the output formatters cliff provides to each command.
By default, the output format is table. Now we choose csv output to run the command list_nets:

```
(quantumv2) list_nets -f csv
"admin_state_up","id","name","status","subnets","tenant_id"
True,"358c43b7-2042-4f5e-848c-b88f441fc71a","test2","ACTIVE","","9d049e4b60b64716978ab
415e6fbd5c0"
True,"5cba94c1-8ed3-44ca-9f8c-a94fd9fc5c6a","test1_0","ACTIVE","","9d049e4b60b64716978a
b415e6fbd5c0"
```

Column selection

We can see -c COLUMN in previous usage output. It can be used to limit the output fields:
(quantumv2) list_nets -f csv -c id -c name
"id","name"
"358c43b7-2042-4f5e-848c-b88f441fc71a","test2"
"5cba94c1-8ed3-44ca-9f8c-a94fd9fc5c6a","test1_0"

please note the difference between -c and -F. -F is implemented by quantum command. it will impact the traffic between server and client. -c is provided by cliff's formatters, which selects the fields on the client side.

Features in V2.0 API

Fields selection

If there are 'fields' in request URL, V2.0 API will extract the list of fields to return.
A sample of such URLs is http://localhost:9696/v2.0/networks.json?fields=id&fields=name

quantumv2 client supports this feature by -F option in known options part and --fields in unknown options part.
For example, quantumv2 -F id list_nets -- --fields name.
Only list_xx and show_xx commands support this feature.

Value filtering

Any other fields except the 'verbose' and 'fields' are used as value filtering.
A sample of such URLs is
http://localhost:9696/v2.0/networks.json?name=test1&name=test2&tag=a
By the current quantum server's sample DB plugin, the filtering has the meaning as SQL clause: name in ['test1', 'test2'].
quantumv2 client supports this feature by any key options in unknown part
For example quantumv2 list_nets -- --name test1 test2 --tag a

Only list_xx and show_xx commands support this feature.

Argument parts of API 2.0 command

In general, arguments divide into three parts:

Known options

These options are following command name. They can be after positional arguments if the command does not support unknow options. Known options are used to represent optional values in API resource. Some options have default value if not specified.

Positional arguments

Positional arguments are for mandatory information for an API resource. There is no optional values for them and must be given in the order.

Unknown options

Unknown options are at the end part of the command line. They must be after a positional argument. If there is no argument for the command, pseudo argument '--' should be used.
To define an unknown option, the format is --optionname [type=int|bool|...] [optionavalue]*.
There can be multiple option values for a certain optionname. When there is no optionvalue given, the option is a bool one and value is true. The type is python built-in type, such as int, bool and float, defaulted to string if not given.
Unknown options are used to provides values for update_command, implement new features of API v2.0. It can also be used to provide information for API extension.
This is a usage for a command which supports unknown options:

```
(quantumv2) create_net -h
usage: create_net [-h] [-f {html,json,shell,table,yaml}] [-c COLUMN]
                  [--variable VARIABLE] [--prefix PREFIX]
                  [--request-format {json,xml}] [--tenant-id tenant-id]
                  [--admin-state-down]
                  name ...
```

Note the "..." after positional argument name, which is the indicator for Unknown options.

Quantumv2 Usage

```
# quantumv2 -H
usage: quantumv2 [--version] [-v] [-q] [-H] [--debug]
                 [--os-auth-url <auth-url>]
                 [--os-auth-strategy <auth-strategy>]
                 [--os-tenant-name <auth-tenant-name>]
                 [--os-username <auth-username>]
                 [--os-password <auth-password>]
```

```
            [--os-region-name <auth-region-name>]
            [--os-quantum-api-version <quantum-api-version>]
            [--os-token <token>] [--os-url <url>]

Command-line interface to the Quantum APIs

optional arguments:
 --version             show program's version number and exit
 -v, --verbose         Increase verbosity of output. Can be repeated.
 -q, --quiet           suppress output except warnings and errors
 -H, --Help            show this help message and exit
 --debug               show tracebacks on errors
  --os-auth-strategy <auth-strategy>
                       Authentication strategy (Env: OS_AUTH_STRATEGY,
                       default keystone). For now, any other value will
                       disable the authentication
 --os-auth-url <auth-url>
                       Authentication URL (Env: OS_AUTH_URL)
 --os-tenant-name <auth-tenant-name>
                       Authentication tenant name (Env: OS_TENANT_NAME)
 --os-username <auth-username>
                       Authentication username (Env: OS_USERNAME)
 --os-password <auth-password>
                       Authentication password (Env: OS_PASSWORD)
 --os-region-name <auth-region-name>
                       Authentication region name (Env: OS_REGION_NAME)
 --os-quantum-api-version <quantum-api-version>
                       QAUNTUM API version, default=2.0 (Env:
                       OS_QUANTUM_API_VERSION)
 --os-token <token>    Defaults to env[OS_TOKEN]
 --os-url <url>        Defaults to env[OS_URL]

Commands for API v2.0:
 create_net     Create a network for a given tenant
 delete_net     Delete a given network
 help           print detailed help for another command
 list_nets      List networks that belong to a given tenant
 show_net       Show information of a given network
 update_net     Update network's information
```

Note:
1. By default, quantumv2 is working on quantum v2.0, which can be changed by

--os-quantum-api-version or env OS_QUANTUM_API_VERSION.
2. -H is used for quantumv2, -h is for sub command. For example, 'quantumv2 list_nets -h' will show the help for the command list_nets, which works the way like 'quantumv2 help list_nets'
3. -v can be used to show HTTP messages between client and server
4. --debug is used to show tracestack when exception raised

Sample usage

Authentication

New quantum client support three ways to auth:
1. no authentication
We can provide value other than keystone to --os-auth-strategy to disable authentication.
2. User and password
With keystone auth strategy, this way needs username, password, tenant name and keystone service URL.
3. Token and quantum URL
With keystone auth strategy, this way just needs a keystone token and quantum service URL.
Let see each of them below:

No authentication

#quantumv2 --os-auth-strategy=et --os-url=http://localhost:9696 list_nets

User and password

#quantumv2 --os-auth-url http://localhost:35357/v2.0 --os-username admin --os-password password --os-tenant-name admin -v list_nets
DEBUG: quantumclient.quantum.v2_0.network.ListNetwork get_data(Namespace(columns=[], filter_specs=[], formatter='table', quote_mode='nonnumeric', request_format='json', show_details=False))
DEBUG: quantumclient.client REQ: curl -i http://localhost:35357/v2.0/tokens -X POST -H "Content-Type: application/json" -H "Accept: application/json" -H "User-Agent: python-quantumclient"

DEBUG: quantumclient.client REQ BODY: {"auth": {"tenantName": "admin", "passwordCredentials": {"username": "admin", "password": "password"}}}

DEBUG: quantumclient.client RESP:{'date': 'Thu, 07 Jun 2012 01:56:33 GMT', 'content-type': 'application/json', 'content-length': '2230', 'status': '200', 'vary': 'X-Auth-Token'}

DEBUG: quantumclient.client RESP BODY:...

DEBUG: quantumclient.client RESP BODY:...

```
+--------------------------------------+----------+
| id                                   | name     |
+--------------------------------------+----------+
| 097dbe2e-11ef-4e96-8b8a-56390232a179 | test9    |
| 4825fd5f-5959-4a1b-94c7-128452e97ff8 | test6    |
| df205add-cd3c-407f-872a-fbbdb1f0da02 | test5    |
| e02c08c9-d685-4a6e-9518-cb078ffacbf1 | private3 |
| fec4871e-a393-413b-ab58-74f750c42c99 | test5    |
+--------------------------------------+----------+
DEBUG: quantumclient.shell clean_up ListNetwork
```

Token and URL way

```
$quantumv2 --os-token ADMIN --os-url http://localhost:9696/ -v list_nets
DEBUG: quantumclient.quantum.v2_0.network.ListNetwork get_data(Namespace(columns=[],
filter_specs=[], formatter='table', quote_mode='nonnumeric', request_format='json',
show_details=False))
DEBUG: quantumclient.quantum.client we are using client: quantumclient.v2_0.client.Client
DEBUG: quantumclient.quantum.v2_0.network.ListNetwork search options: {}
DEBUG: quantumclient.client REQ: curl -i http://localhost:9696/v2.0/networks.json? -X GET -H
"User-Agent: python-quantumclient" -H "Content-Type: application/json" -H "Accept:
application/json" -H "X-Auth-Token: ADMIN"

DEBUG: quantumclient.client RESP:{'date': 'Thu, 07 Jun 2012 02:00:59 GMT', 'status': '200',
'content-length': '1671', 'content-type': 'application/json', 'content-location':
'http://localhost:9696/v2.0/networks.json?'}

DEBUG: ...

+--------------------------------------+----------+
| id                                   | name     |
+--------------------------------------+----------+
| 097dbe2e-11ef-4e96-8b8a-56390232a179 | test9    |
| df205add-cd3c-407f-872a-fbbdb1f0da02 | test5    |
| e02c08c9-d685-4a6e-9518-cb078ffacbf1 | private3 |
| fec4871e-a393-413b-ab58-74f750c42c99 | test5    |
+--------------------------------------+----------+
DEBUG: quantumclient.shell clean_up ListNetwork
```

Authn wrap up

user and password way will authn the user, while the token will not. But the quantum server will check the token's validity against the identity server.

ENV for quantumv2 authn

```
export OS_USERNAME=admin
export OS_PASSWORD=password
export OS_TENANT_NAME=admin
export OS_AUTH_URL=http://localhost:5000/v2.0

Or
export OS_TOKEN=ADMIN
export OS_URL=http://localhost:9696/

Or
export OS_AUTH_STRATEGY=noauth
export OS_URL=http://localhost:9696/
```

Create network

Show the help of create_net command

```
# quantumv2 -v create_net -h
usage: quantumv2 create_net [-h] [-f {html,json,shell,table,yaml}] [-c COLUMN]
                            [--variable VARIABLE] [--prefix PREFIX]
                            [--request-format {json,xml}]
                            [--tenant-id tenant-id] [--admin-state-down]
                            name ...

Create a network for a given tenant Sample create_net --tenant-id xxx --admin-
state-up <net_name> --tag x y

positional arguments:
  net-name              Name of network to create
  value_specs           extra values for the network:--field1
                        [type=int|bool|...] value [--field2
                        [type=int|bool|...] value ...]

optional arguments:
  -h, --help            show this help message and exit
  --request-format {json,xml}
                        the xml or json request format
  --tenant-id tenant-id
                        the owner tenant ID
```

```
  --admin-state-down    Set Admin State Up to false

output formatters:
  output formatter options

  -f {html,json,shell,table,yaml}, --format {html,json,shell,table,yaml}
                        the output format to use, defaults to table
  -c COLUMN, --column COLUMN
                        specify the column(s) to include, can be repeated

shell formatter:
  Print values in a format a UNIX shell can parse (variable="value")

  --variable VARIABLE   specify the variable(s) to include, can be repeated
  --prefix PREFIX       add a prefix to all variable names
```

Create network

```
# quantumv2 -v create_net test11
DEBUG: quantumclient.quantum.v2_0.network.CreateNetwork
get_data(Namespace(admin_state_down=True, columns=[], formatter='table',
net_name='test11', prefix='', request_format='json', tenant_id=None, value_specs=[],
variables=[]))
DEBUG: quantumclient.quantum.client we are using client: quantumclient.v2_0.client.Client
DEBUG: quantumclient.client REQ: curl -i http://localhost:9696/v2.0/networks.json -X POST -H
"User-Agent: python-quantumclient" -H "Content-Type: application/json" -H "Accept:
application/json" -H "X-Auth-Token: ADMIN"

DEBUG: quantumclient.client REQ BODY: {"network": {"name": "test11", "admin_state_up":
true}}

DEBUG: quantumclient.client RESP:{'date': 'Thu, 07 Jun 2012 02:11:30 GMT', 'status': '201',
'content-length': '139', 'content-type': 'application/json'}

DEBUG: quantumclient.client RESP BODY:{"network": {"subnets": [], "op_status": "ACTIVE",
"id": "214d622d-0850-4a20-b1bb-242bb1fbf3f9", "name": "test11", "admin_state_up": true}}

Created a new Virtual Network:
+----------------+--------------------------------------+
| Field          | Value                                |
+----------------+--------------------------------------+
| admin_state_up | True                                 |
```

```
| id            | 214d622d-0850-4a20-b1bb-242bb1fbf3f9 |
| name          | test11                               |
| op_status     | ACTIVE                               |
| subnets       | []                                   |
+---------------+--------------------------------------+
DEBUG: quantumclient.shell clean_up CreateNetwork
```

Create network with any extra fields

```
# quantumv2 -v create_net test12 --op_status DOWN --tag x y --anyfield value
DEBUG: quantumclient.quantum.v2_0.network.CreateNetwork
get_data(Namespace(admin_state_down=True, columns=[], formatter='table',
net_name='test12', prefix='', request_format='json', tenant_id=None, value_specs=['--op_status',
'DOWN', '--tag', 'x', 'y', '--anyfield', 'value'], variables=[]))
DEBUG: quantumclient.quantum.client we are using client: quantumclient.v2_0.client.Client
DEBUG: quantumclient.client REQ: curl -i http://localhost:9696/v2.0/networks.json -X POST -H
"User-Agent: python-quantumclient" -H "Content-Type: application/json" -H "Accept:
application/json" -H "X-Auth-Token: ADMIN"

DEBUG: quantumclient.client REQ BODY: {"network": {"anyfield": "value", "tag": ["x", "y"],
"op_status": "DOWN", "name": "test12", "admin_state_up": true}}

DEBUG: quantumclient.client RESP:{'date': 'Thu, 07 Jun 2012 02:13:26 GMT', 'status': '201',
'content-length': '139', 'content-type': 'application/json'}

DEBUG: quantumclient.client RESP BODY:{"network": {"subnets": [], "op_status": "ACTIVE",
"id": "c6aa7ab3-1d4e-4917-b41d-f3c741d9b788", "name": "test12", "admin_state_up": true}}

Created a new Virtual Network:
+---------------+--------------------------------------+
| Field         | Value                                |
+---------------+--------------------------------------+
| admin_state_up | True                                |
| id            | c6aa7ab3-1d4e-4917-b41d-f3c741d9b788 |
| name          | test12                               |
| op_status     | ACTIVE                               |
| subnets       | []                                   |
+---------------+--------------------------------------+
DEBUG: quantumclient.shell clean_up CreateNetwork
```

Note:
1.we can see that the the last part of command line '--op_status DOWN --tag x y --anyfield value' is converted into res body.
2. we must put extra fields to the end of the command line.
3. we can create network for other tenant by the option --tenant-id

List nets

Usage

```
# quantumv2 list_nets -h
usage: quantumv2 list_nets [-h] [-f {csv,html,json,table,yaml}] [-c COLUMN]
                           [--quote {all,minimal,none,nonnumeric}]
                           [--request-format {json,xml}] [-D]
                           ...

List networks that belong to a given tenant Sample: list_nets -D --
--name=test4 --tag a b

positional arguments:
  filter_specs          filters options: --field1 [type=int|bool|...] value
                        [--field2 [type=int|bool|...] value ...]

optional arguments:
  -h, --help            show this help message and exit
  --request-format {json,xml}
                        the xml or json request format
  -D, --show-details    show detailed info

output formatters:
  output formatter options

  -f {csv,html,json,table,yaml}, --format {csv,html,json,table,yaml}
                        the output format to use, defaults to table
  -c COLUMN, --column COLUMN
                        specify the column(s) to include, can be repeated

CSV Formatter:
  --quote {all,minimal,none,nonnumeric}
                        when to include quotes, defaults to nonnumeric
```

List network without extra field

```
# quantumv2 list_nets
+--------------------------------------+----------+
| id                                   | name     |
+--------------------------------------+----------+
| 097dbe2e-11ef-4e96-8b8a-56390232a179 | test9    |
| 214d622d-0850-4a20-b1bb-242bb1fbf3f9 | test11   |
| 41c27b8e-d67c-47cb-abba-3239e1d66ddd | test10   |
| 4825fd5f-5959-4a1b-94c7-128452e97ff8 | test6    |
| 5ca42d78-a5e5-495b-b6de-96abc2b4420e | test8    |
| 5e61de4c-ee6f-4a22-a495-25aa7a43d5a2 | test6    |
| 6c3df337-b585-4c0f-b80e-e47a56bdad83 | test7    |
| 73af0539-7b89-457b-9b12-c6c3690ae541 | test8    |
| 768f9cff-fa8e-460d-a656-c71dcfac6cad | test9    |
| 7a9e7a1c-343e-407b-9494-75a7a1f85456 | test8    |
| 9392879d-22d6-490e-a2b6-797384dda587 | test7    |
| a1bd80dc-c1a9-4857-961a-2993fa9a9c49 | newname  |
| c6aa7ab3-1d4e-4917-b41d-f3c741d9b788 | test12   |
| d751d616-e02d-4b93-b821-2234463547fe | test5    |
| df205add-cd3c-407f-872a-fbbdb1f0da02 | test5    |
| e02c08c9-d685-4a6e-9518-cb078ffacbf1 | private3 |
| fec4871e-a393-413b-ab58-74f750c42c99 | test5    |
+--------------------------------------+----------+
```

List network with extra fields

```
# quantumv2 -v list_nets -D -- --tag x y --key1 value1 --intkey type=int 2
DEBUG: quantumclient.quantum.v2_0.network.ListNetwork get_data(Namespace(columns=[],
filter_specs=['--', '--tag', 'x', 'y', '--key1', 'value1', '--intkey', 'type=int', '2'], formatter='table',
quote_mode='nonnumeric', request_format='json', show_details=True))
DEBUG: quantumclient.quantum.client we are using client: quantumclient.v2_0.client.Client
DEBUG: quantumclient.quantum.v2_0.network.ListNetwork search options: {'key1': 'value1',
'tag': ['x', 'y'], 'intkey': 2}
DEBUG: quantumclient.client REQ: curl -i
http://localhost:9696/v2.0/networks.json?key1=value1&tag=x&tag=y&verbose=True&intkey=2 -X
GET -H "User-Agent: python-quantumclient" -H "Content-Type: application/json" -H "Accept:
application/json" -H "X-Auth-Token: ADMIN"

DEBUG: quantumclient.client RESP:{'date': 'Thu, 07 Jun 2012 02:19:33 GMT', 'status': '200',
'content-length': '2182', 'content-type': 'application/json', 'content-location':
'http://localhost:9696/v2.0/networks.json?key1=value1&tag=x&tag=y&verbose=True&intkey=2'}
```

DEBUG: quantumclient.client RESP BODY:{"networks": [{"subnets": [], "op_status": "DOWN", "id": "097dbe2e-11ef-4e96-8b8a-56390232a179", "name": "test9", "admin_state_up": true}, {"subnets": [], "op_status": "ACTIVE", "id": "214d622d-0850-4a20-b1bb-242bb1fbf3f9", "name": "test11", "admin_state_up": true}, {"subnets": [], "op_status": "ACTIVE", "id": "41c27b8e-d67c-47cb-abba-3239e1d66ddd", "name": "test10", "admin_state_up": true}, {"subnets": [], "op_status": "ACTIVE", "id": "4825fd5f-5959-4a1b-94c7-128452e97ff8", "name": "test6", "admin_state_up": true}, {"subnets": [], "op_status": "ACTIVE", "id": "5ca42d78-a5e5-495b-b6de-96abc2b4420e", "name": "test8", "admin_state_up": false}, {"subnets": [], "op_status": "ACTIVE", "id": "5e61de4c-ee6f-4a22-a495-25aa7a43d5a2", "name": "test6", "admin_state_up": true}, {"subnets": [], "op_status": "ACTIVE", "id": "6c3df337-b585-4c0f-b80e-e47a56bdad83", "name": "test7", "admin_state_up": true}, {"subnets": [], "op_status": "ACTIVE", "id": "73af0539-7b89-457b-9b12-c6c3690ae541", "name": "test8", "admin_state_up": false}, {"subnets": [], "op_status": "ACTIVE", "id": "768f9cff-fa8e-460d-a656-c71dcfac6cad", "name": "test9", "admin_state_up": true}, {"subnets": [], "op_status": "ACTIVE", "id": "7a9e7a1c-343e-407b-9494-75a7a1f85456", "name": "test8", "admin_state_up": false}, {"subnets": [], "op_status": "ACTIVE", "id": "9392879d-22d6-490e-a2b6-797384dda587", "name": "test7", "admin_state_up": true}, {"subnets": [], "op_status": "ACTIVE", "id": "a1bd80dc-c1a9-4857-961a-2993fa9a9c49", "name": "newname", "admin_state_up": true}, {"subnets": [], "op_status": "ACTIVE", "id": "c6aa7ab3-1d4e-4917-b41d-f3c741d9b788", "name": "test12", "admin_state_up": true}, {"subnets": [], "op_status": "ACTIVE", "id": "d751d616-e02d-4b93-b821-2234463547fe", "name": "test5", "admin_state_up": true}, {"subnets": [], "op_status": "ACTIVE", "id": "df205add-cd3c-407f-872a-fbbdb1f0da02", "name": "test5", "admin_state_up": true}, {"subnets": [], "op_status": "ACTIVE", "id": "e02c08c9-d685-4a6e-9518-cb078ffacbf1", "name": "private3", "admin_state_up": true}, {"subnets": [], "op_status": "ACTIVE", "id": "fec4871e-a393-413b-ab58-74f750c42c99", "name": "test5", "admin_state_up": true}]}

```
+--------------------------------------+----------+-----------+----------------+
| id                                   | name     | op_status | admin_state_up |
+--------------------------------------+----------+-----------+----------------+
| 097dbe2e-11ef-4e96-8b8a-56390232a179 | test9    | DOWN      | True           |
| 214d622d-0850-4a20-b1bb-242bb1fbf3f9 | test11   | ACTIVE    | True           |
| 41c27b8e-d67c-47cb-abba-3239e1d66ddd | test10   | ACTIVE    | True           |
| 4825fd5f-5959-4a1b-94c7-128452e97ff8 | test6    | ACTIVE    | True           |
| 5ca42d78-a5e5-495b-b6de-96abc2b4420e | test8    | ACTIVE    | False          |
| 5e61de4c-ee6f-4a22-a495-25aa7a43d5a2 | test6    | ACTIVE    | True           |
| 6c3df337-b585-4c0f-b80e-e47a56bdad83 | test7    | ACTIVE    | True           |
| 73af0539-7b89-457b-9b12-c6c3690ae541 | test8    | ACTIVE    | False          |
| 768f9cff-fa8e-460d-a656-c71dcfac6cad | test9    | ACTIVE    | True           |
| 7a9e7a1c-343e-407b-9494-75a7a1f85456 | test8    | ACTIVE    | False          |
| 9392879d-22d6-490e-a2b6-797384dda587 | test7    | ACTIVE    | True           |
| a1bd80dc-c1a9-4857-961a-2993fa9a9c49 | newname  | ACTIVE    | True           |
```

```
| c6aa7ab3-1d4e-4917-b41d-f3c741d9b788 | test12   | ACTIVE    | True          |
| d751d616-e02d-4b93-b821-2234463547fe | test5    | ACTIVE    | True          |
| df205add-cd3c-407f-872a-fbbdb1f0da02 | test5    | ACTIVE    | True          |
| e02c08c9-d685-4a6e-9518-cb078ffacbf1 | private3 | ACTIVE    | True          |
| fec4871e-a393-413b-ab58-74f750c42c99 | test5    | ACTIVE    | True          |
+--------------------------------------+----------+-----------+---------------+
DEBUG: quantumclient.shell clean_up ListNetwork
```

Note:
1. extra field can have python built in type.
2. list_net should use '--' pseudo argument before extra field
3. the extra fields are appended to GET URL

Update network

Usage

```
# quantumv2 update_net -h
usage: quantumv2 update_net [-h] [--request-format {json,xml}] net-id ...

Update network's information Sample: update_net <net_id> --name=test
--admin_state_up type=bool True

positional arguments:
  net-id                ID of network to update
  value_specs           new values for the network--field1 [type=int|bool|...]
                        value [--field2 [type=int|bool|...] value ...]

optional arguments:
  -h, --help            show this help message and exit
  --request-format {json,xml}
                        the xml or json request format
```

Update network

```
# quantumv2 -v update_net 097dbe2e-11ef-4e96-8b8a-56390232a179 --name test9_0
--op_status=ACTIVE --admin_state_up type=bool false --tag xx yy a=b --anykey value
DEBUG: quantumclient.quantum.v2_0.network.UpdateNetwork
run(Namespace(net_id='097dbe2e-11ef-4e96-8b8a-56390232a179', request_format='json',
value_specs=['--name', 'test9_0', '--op_status=ACTIVE', '--admin_state_up', 'type=bool', 'false',
'--tag', 'xx', 'yy', 'a=b', '--anykey', 'value']))
DEBUG: quantumclient.quantum.client we are using client: quantumclient.v2_0.client.Client
```

DEBUG: quantumclient.client REQ: curl -i http://localhost:9696/v2.0/networks/097dbe2e-11ef-4e96-8b8a-56390232a179.json PUT -H "User-Agent: python-quantumclient" -H "Content-Type: application/json" -H "Accept: application/json" -H "X-Auth-Token: ADMIN"

DEBUG: quantumclient.client REQ BODY: {"network": {"name": "test9_0", "op_status": "ACTIVE", "admin_state_up": false, "anykey": "value", "tag": ["xx", "yy", "a=b"], "id": "097dbe2e-11ef-4e96-8b8a-56390232a179"}}

DEBUG: quantumclient.client RESP:{'date': 'Thu, 07 Jun 2012 02:25:32 GMT', 'status': '200', 'content-length': '141', 'content-type': 'application/json'}

DEBUG: quantumclient.client RESP BODY:{"network": {"subnets": [], "op_status": "ACTIVE", "id": "097dbe2e-11ef-4e96-8b8a-56390232a179", "name": "test9_0", "admin_state_up": false}}

Updated Network: 097dbe2e-11ef-4e96-8b8a-56390232a179
DEBUG: quantumclient.shell clean_up UpdateNetwork

Note:
1. This command use extra fields feature to specify the values. This way, we can specify any key value pairs to update a network
2. the information are converted into req body.
3. multiple values for a certain key is stored into a list

Delete network

usage

quantumv2 delete_net -h
usage: quantumv2 delete_net [-h] [--request-format {json,xml}] net-id

Delete a given network Sample: delete_net <net_id>

positional arguments:
  net-id   ID of network to delete

optional arguments:
  -h, --help   show this help message and exit
  --request-format {json,xml}
                        the xml or json request format

Delete a network sample

```
# quantumv2 -v delete_net d751d616-e02d-4b93-b821-2234463547fe
DEBUG: quantumclient.quantum.v2_0.network.CreateNetwork
run(Namespace(net_id='d751d616-e02d-4b93-b821-2234463547fe', request_format='json'))
DEBUG: quantumclient.quantum.client we are using client: quantumclient.v2_0.client.Client
DEBUG: quantumclient.client REQ: curl -i
http://localhost:9696/v2.0/networks/d751d616-e02d-4b93-b821-2234463547fe.json DELETE -H
"User-Agent: python-quantumclient" -H "Content-Type: application/json" -H "Accept:
application/json" -H "X-Auth-Token: ADMIN"

DEBUG: quantumclient.client RESP:{'date': 'Thu, 07 Jun 2012 02:30:34 GMT', 'status': '204',
'content-length': '4'}

DEBUG: quantumclient.client RESP BODY:

Deleted Network: d751d616-e02d-4b93-b821-2234463547fe
DEBUG: quantumclient.shell clean_up DeleteNetwork
```

Note:
1. unlike other command, delete_net has no extra fields feature.