Galaxy API
Galaxy's REST Overview

- Uses generated API keys for per-user authentication
  - No username/password
  - No credential caching, yet (so not really REST)
- Request parameters and responses are in JSON (JavaScript Object Notation)
- Maintains security
- Enable with `enable_api = True` in config (enabled by default)
API Interactions with Galaxy

Galaxy's REST API

$GALAXY_HOME/scripts/api

python lib: blend

JavaScript lib

Galaxy’s REST API

Galaxy
Bare Bones Galaxy API

- create (POST)
- display (GET)
- update (PUT)
- delete (DELETE)
Making the Calls

Wrapper methods exist (in `/scripts/api/`) to make the API calls easier:

```
./{action}.py <api key> http://<ip>/api/{module}/[unit] [args]
```

- **action**: create | display | update | delete

- **api_key**: obtained from the UI

- **module**: datasets | forms | histories | libraries | permissions | quotas | requests | roles | samples | tools | users | visualizations | workflows

- **unit**: dataset_id / history_id / library_id / …

- **args**: name / key-value pair / …
Install a few things in the VM

```
sudo apt-get install gnome-terminal ipython python python-pip
```
Hands on time
Workshop tasks

- Run a workflow
- Create an API script; script for auto data move
- curl
- Galaxy
blend

- A python library for interacting with Galaxy’s API
- And CloudMan
Request compute infrastructure
Manipulate compute infrastructure
Upload data and run analyses
Docs and examples
Test
Distribute

Automate repetitive tasks
Blend is a Python (2.6 or higher) library for interacting with BioCloudCentral.org, CloudMan, and Galaxy’s API. Conceptually, it makes it possible to script and automate the process of cloud infrastructure provisioning and scaling, as well as running of analyses within Galaxy. In reality, it makes it possible to do things like this:

- Create a CloudMan compute cluster, via an API and directly from your local machine:

```python
from blend.cloudman.launch import CloudManLaunch
cml = CloudManLaunch('your cloud access key','your cloud secret key')
cml.launch('Blend CloudMan','ami-ID','ml.small','password')
cml.get_status()
```

- Manipulate your CloudMan instance and react to the current needs:

```python
from blend.cloudman import CloudMan
cm = CloudMan('instance IP','password')
cm.initialize(type='Galaxy')
cm.add_nodes(3)
cluster_status = cm.get_status()
cm.remove_nodes(2)
```

- Interact with Galaxy via a straightforward API:

```python
from blend.galaxy import GalaxyInstance
gi = GalaxyInstance('Galaxy IP', key='your API key')
libs = gi.libraries.get_libraries()
gi.workflows.show_workflow('workflow ID')
gi.workflows.run_workflow('workflow ID', input_dataset_map)
```

**Note**

Although this library allows you to blend these three services into a cohesive unit, the library itself can be used with any single service irrespective of the set. For example, you can use it to just manipulate CloudMan cluster sets.
Install blend

$ sudo pip install blend-lib

$ ipython
Working with Galaxy’s API

```python
from blend.galaxy import GalaxyInstance

# Create a Galaxy instance
gi = GalaxyInstance('Galaxy IP', key='API key')

# Get libraries
gi.libraries.get_libraries()

# Get workflows
w = gi.workflows.get_workflows()[0]

# Show workflow
gi.workflows.show_workflow(w['id'])

# Get histories
gi.histories.get_histories(name='Unnamed history')
```
1. Create a data library and upload some data into it
2. Create a history
3. Execute a workflow
Import a workflow

Use the Galaxy UI:

Workflows -> Import Workflow

http://tinyurl.com/gcc-wf
from blend.galaxy import GalaxyInstance
# Define a connection to an instance of Galaxy
gi = GalaxyInstance('http://127.0.0.1:8085', 'c18666899798c5ad233c511cd2d66c2d')
# Create a data library
l = gi.libraries.create_library('WS6')
# Upload some data to the data library
d1 = gi.libraries.upload_file_from_url(l[0]['id'], 'http://tinyurl.com/gcc-exons')
d2 = gi.libraries.upload_file_from_url(l[0]['id'], 'http://tinyurl.com/gcc-snps')
# Create a history
gi.histories.create_history('Run 6')
# Get information on how to run a workflow
ws = gi.workflows.get_workflows()
w = gi.workflows.show_workflow(ws[0]['id'])
# {u'id': u'93ab6bbd094e1dcd',
#  u'inputs': {u'1': {u'label': u'Input Dataset', u'value': u''},
#  u'3': {u'label': u'Input Dataset', u'value': u''}},
#  u'name': u'Demo workflow',
#  u'url': u'/api/workflows/93ab6bbd094e1dcd'}
dataset_map = {'1': {'id':d1[0]['id'], 'src':'ld'},
  '3': {'id':d2[0]['id'], 'src':'ld'}}
wr = gi.workflows.run_workflow(w['id'], dataset_map, h['id'])
hl = gi.histories.get_histories()
wy = gi.histories.show_history(hl[0]['id'], contents=True)
Blend’s state: *public comment*

- Blend wraps Galaxy’s existing API
  - It does not add any functionality but simply exposes it as a python library
- About half of Galaxy’s API have been wrapped
  - What’s not yet there is the sequencer integration & forms API
- Read the docs: blend.readthedocs.org