The Galaxy API

(or, how to do things)

What’s an API?

- Links (URLs) no longer simply load a page or complete a form.
- Modern web apps use urls as functions.
- We’re building galaxy on top of the API almost exclusively going forward.

What's it good for?

UI: good for exploring data and experimenting

API scripting: good for automation of repeated tasks

UI: good for those not comfortable with command line

API scripting: good for power users and integration with non-Galaxy resources

Both: allows the sharing/collaborating/recording of analyses using a common resource (Galaxy)

REST?

Create : HTTP POST + resource URL

Read : GET + resource URL

Update : PUT + resource URL

Delete : DELETE* + resource URL

stateless:

nothing is stored on the server between requests and each request must contain all the info the server needs

Resources, URLs, and parameters

Galaxy has many resources:

- datasets
- tools
- workflows, histories, libraries, etc.

Each resource maps to an api URL:

- api/datasets
- api/tools
- api/workflows, api/histories, api/libraries, etc.

Resources, URLs, and parameters

We call an API function using: HTTP VERB + noun + extras:

GET /api/tools/sort1?io_details=True -> detailed tool info

POST /api/histories { “name”: “new” } -> create a history named “new”

PUT /api/histories/<some id> { “published”: true } -> publish a history

DELETE /api/histories/<id>/contents/<id> -> delete an history’s dataset

URL params (e.g. ids) vs. query params (‘?...’)

POST/PUT params as JSON

How to access the API

- When the UI accesses the API, session authentication is used.
- When external callers (such as scripts) access, an API key must be passed as well.
- API keys are only available via the UI.

JavaScript Object Notation (JSON)

A concise way of building data structures

A medium of interchange for the API

Fully described at: json.org

!Watch those ‘quotation’ “marks”

Security

Keep keys in a secure location:

- they’re the same as a username and password

SSL recommended:

- http://localhost?key=foo could be sniffed
- https://localhost?key=foo safer

Errors

HTTP status codes:

200 ok, 400 your error, 500 server error

Galaxy error codes:

lib/galaxy/exceptions/error_codes.json

A work in progress

Error handling: for the public & your future self

Analysis Basics

If you're not familiar with these Galaxy resources:

histories, datasets, workflows, tools

let us know now! (we're happy to give an overview)

Analysis Exercise #1

1. Create a history
2. upload a file to a dataset
3. import a workflow
4. run the workflow on the data

Analysis Exercise #1 - create a history

GET + histories = list histories

POST + histories = create a history

GET + histories + id = detailed history info

Analysis Exercise #1 - upload a file

POST + tools + upload1 = run the upload tool

GET + history + ‘/contents/’ + hda = did it work?

(Histories and libraries contain datasets)

Analysis Exercise #1 - import a workflow

GET + workflows = list my workflows

Nothing! Let's import one:

    data/Bed_interval_lengths.ga

POST + workflows/upload + workflow = import

Analysis Exercise #1 - run a workflow

GET + workflows + id = detailed info

(find the input steps) { “inputs”: … }

POST + workflows + data = run a workflow

Analysis Exercise #1 - extras

- viewing the resulting data in a number of formats
- downloading the results to the filesystem
- publishing the history

Analysis Exercise extras - viewing data

GET + dataset + id + ...how to view = data json

Data providers: do not alter data, provide a view

- line : return data as strings
- column : return data as lists/arrays
- dict : return data as dictionaries

Analysis Exercise extras - downloading

GET + dataset + id + display = download

Returns as text -> redirect to file

Analysis Exercise extras - publishing

Update history to published = true

PUT + histories + id + { “published”: true } =

make the history accessible and publish

Analysis Exercise #1

Questions?

(break time)

Proxy Configuration for External Auth

```
location / {
    auth_ldap_require valid_user;
    auth_ldap "LDAP Auth Source Description";
    proxy_set_header REMOTE_USER $remote_user;
    proxy_pass  http://galaxy_app;
    proxy_set_header   X-Forwarded-Host $host;
    proxy_set_header   X-Forwarded-For  $proxy_add_x_forwarded_for;
    proxy_set_header   X-URL-SCHEME https;
  ...
}

location  /api {
    proxy_set_header REMOTE_USER $remote_user;
    proxy_pass  http://galaxy_app;
    proxy_set_header   X-Forwarded-Host $host;
    proxy_set_header   X-Forwarded-For  $proxy_add_x_forwarded_for;
}
```

For API access, set REMOTE_USER if available so Galaxy session based requests are let through.
If REMOTE_USER is not available pass the request through and let Galaxy determine if a key is present and valid.

Administrative Usage, Extra Setup

User impersonation with run_as

```
# Optional list of email addresses of API users who can make calls on behalf of
# other users
api_allow_run_as=foo@foo.com

# Master key that allows many API admin actions to be used without actually
# having a defined admin user in the database/config. Only set this if you need
# to bootstrap Galaxy, you probably do not want to set this on public servers.
master_api_key=MASTERLOCK

enable_quotas = True
```

Exercise - Tool Installation

The Task: Given a yaml file with several tools listed, install them all to a local galaxy.

```
---
toolshed.g2.bx.psu.edu:
  devteam:
    - bam_to_sam
    - ctd_batch
```

Bonus Points: Query the instance tools list to see if the tool exists first, and only attempt install if it doesn't.

Exercise - Scaffold a new User

The Task: Write a single script to create a new user account, set an initial quota, and upload some starter data.

```
python scaffold_user.py <username> <email> <password>
```

Exercise - Disk Usage and Job Report

The Task: Write a script that prints out how much disk space users are using, and how many currently running jobs they have.

Exercise - Galactic Dropbox

The Task: We want to build a program that monitors a ‘drop location’ on the filesystem. Anything that shows up here will be linked (upload, but no copy) into a data library.

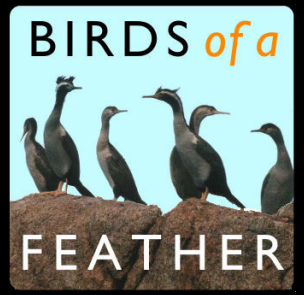
Bonus Points: If a file is removed from disk, remove it from the library.

BioBlend

BioBlend is a Python (2.6 or 2.7) library for interacting with Galaxy and Cloudman APIs.

Provides a nice object layer on top of the pure REST invocations.

Doing the Branch, Release and Merge Waltz



Monday, June 30, 6:15pm
Salon A

http://bit.ly/gcc2014mergebof

We will focus on branching and release management with regard to existing instances which implement customized code within Galaxy. This may create huge challenges in the future, especially for instances in production which require a lot of maintenance and which run older versions of Galaxy. All Clouds and Clusters which require multiple extensions like:

- huge file management (upload, etc)
- authentication issues
- cluster/cloud connectivity
- And the customization of these issues is not easy and

Break @ 3:00 - 3:30

Drinks and snacks will be available during the break, and in all Training Day Rooms after this workshop.