BOSC/Broad Interoperability Hackathon Goals

This document helps brainstorm development goals for the BOSC/Broad 2013 Hackathon. All ideas are welcome.

- Shared interoperability APIs -- Projects like Galaxy and GenomeSpace provide REST based APIs and higher level language toolkits to perform functionality like file transfer and analysis processing. What shared functionality do they have that could be extracted into higher level too that support multiple backends? Could we put together a rough spec that other software could use to implement file input/download and processing?
    - Galaxy API: https://galaxy-dist.readthedocs.org/en/latest/lib/galaxy.webapps.galaxy.api.html
    - GenomeSpace API: http://www.genomespace.org/support/api/restful-access-to-dm
    - BaseSpace API: https://developer.basespace.illumina.com/docs
    - 23andMe API: https://api.23andme.com/docs/
    -
    - Current wrappers
        - Galaxy Python: https://github.com/afgane/bioblend
        - Galaxy Java: https://github.com/jmchilton/blend4j
        - Galaxy Clojure: https://github.com/chapmanb/clj-blend
        - GenomeSpace Java: http://www.genomespace.org/support/api/cdk
        - GenomeSpace Clojure: https://github.com/chapmanb/clj-genomespace
        - BaseSpace Java: https://github.com/heuermh/basespace-java-sdk
        - 23andMe Java: https://github.com/heuermh/personal-genome-client
- Supporting general cloud backends with similar transfer APIs. One important component of analysis pipelines is storing files along with associated reproducibility metadata: file provenance, tool versions used, analysis parameters. Could the interoperability APIs be extended/generalized to support transfer to key-value-plus-metadata style databases?
    - Amazon S3: http://aws.amazon.com/s3/
    - boto, Python interface to Amazon: https://github.com/boto/boto
    - jclouds, Java interface to Amazon: http://www.jclouds.org/
    - Riak S3 compatible storage: https://github.com/basho/riak_cs
- Investigate use of Evented APIs for notification and data tracking between services. http://www.eventedapi.org/
- Consider use of a common package definition (similar to node.js plugins package.json: https://npmjs.org/doc/json.html), to define plugins / pluggable components for various platforms.

Morning Discussion -- Apr 7

General goals:

- Develop clients to existing software, provide generic API and functionality
- Main use case is building connected biological software
- Provide recommendations for server side REST APIs to provide this functionality so existing clients will work with new services

API goals:

- Authentication
- Discovery + query of file collections/manifest of files and data
- Mechanisms to push and pull files and data, along with interacting with remote files without pulling (BAMs + indexes)
- metadata and context queries for files and collections/manifests
- file provenance and tracking
- pubsub or eventing APIs to allow registering for events and retrieve push information about changes that can trigger

Genome connector APIs

Notes:

List/getFile should also include a version that accepts a URL as parameter
List vs. list meta or list file sets
getMetadata/meta only works on a file set
Get metadata for e.g. an index should return metadata about the entire file set it belongs to
  name, size, date, checksum
  pointers to tightly connected files, loosely connected files
  server can store metadata in e.g. .metadata.json files
  metadata should have a stable identifier/canonical name/GUID
  link/reference section might be expanded to support provenance
List should return file sets, not individual files
Single file versus collection of file, not directory vs. file
Manifest is at the directory devel
Find/search files that match these criteria
Put/post dialog
  two-step, first announce intent to upload file, with metadata
  second step, actually send/upload with/without multipart
    here's where to send the files

Notes about method naming:

discover
authenticate

listFileSets (get one entry for each set of e.g. bam and bai)
listFiles (list each file)

meta(fileSet)
meta(url) (url could refer to a fileset or a file), if file - infer fileset

get(fileobject)
get(uri) -> “smart” overload, infers metat information, creates file object etc.

getContainer

put(container, filehandle)

discovery protocol:
browsing root uri
get uri
put uri
auth uri

Python API
https://github.com/roryk/python-gcon

Clojure API
https://github.com/chapmanb/clj-gcon/blob/master/src/gcon/core.clj

Java API
https://github.com/heuermh/java-gcon/blob/master/gcon-api/src/main/java/com/github/heuermh/gcon/GenomeConnector.java

https://github.com/heuermh/java-gcon/blob/master/gcon-api/src/main/java/com/github/heuermh/gcon/GenomeConnectorClient.java

Capability list

GenomeSpace

login: uses token with basic authentication

Galaxy

Auth: api key, attached to each request ?key=XXX
Discovery:
Metadata:

BaseSpace

Authentication uses OAuth 2.0; app developers must register their apps to obtain a client_id. This client_id is used to retrieve an access_token which is then passed along with API calls.

https://developer.basespace.illumina.com/docs/content/documentation/authentication/obtaining-access-tokens#OAuth_flow_for_non-Web_based_applications

REST APIs:

File discovery

List files per run, per sample, or per appresult
https://developer.basespace.illumina.com/docs/content/documentation/rest-api/api-reference#GET%3a_runs%2f{id}%2ffiles

https://developer.basespace.illumina.com/docs/content/documentation/rest-api/api-reference#GET%3a_samples%2f%7bId%7d%2ffiles

https://developer.basespace.illumina.com/docs/content/documentation/rest-api/api-reference#GET%3a_appresults%2f%7bId%7d%2ffiles

Get file metadata
https://developer.basespace.illumina.com/docs/content/documentation/rest-api/api-reference#GET%3a_files%2f%7bId%7d

Get file content
https://developer.basespace.illumina.com/docs/content/documentation/rest-api/api-reference#GET%3a_files%2f{Id}%2fcontent

Multi-part upload APIs
https://developer.basespace.illumina.com/docs/content/documentation/app-integration/multipartfileupload

Java APIs:

File discovery

List files per run, per sample, or per appresult
(not available in public javadocs)
file:///Users/heuermh/working/basespace-java-sdk/target/site/apidocs/com/illumina/basespace/BaseSpaceSession.html#getFiles(com.illumina.basespace.Run,%20com.illumina.basespace.FileFetchParams)

(not available in public javadocs)
file:///Users/heuermh/working/basespace-java-sdk/target/site/apidocs/com/illumina/basespace/BaseSpaceSession.html#getFiles(com.illumina.basespace.Sample,%20com.illumina.basespace.FileFetchParams)

(not available in public javadocs)
file:///Users/heuermh/working/basespace-java-sdk/target/site/apidocs/com/illumina/basespace/BaseSpaceSession.html#getFiles(com.illumina.basespace.AppResult,%20com.illumina.basespace.FileFetchParams)

Get file metadata
http://basespace.github.io/basespace-java-sdk/com/illumina/basespace/BaseSpaceSession.html#getFile(java.lang.String)

There is also a FileMetaData class but no API call on BaseSpaceSession to retrieve such
(not available in public javadocs)
file:///Users/heuermh/working/basespace-java-sdk/target/site/apidocs/com/illumina/basespace/FileMetaData.html

Get file as input stream
(not available in public javadocs)
file:///Users/heuermh/working/basespace-java-sdk/target/site/apidocs/com/illumina/basespace/BaseSpaceSession.html#getFileInputStream(com.illumina.basespace.File)

Get range of file as input stream
(not available in public javadocs)
file:///Users/heuermh/working/basespace-java-sdk/target/site/apidocs/com/illumina/basespace/BaseSpaceSession.html#getFileInputStream(com.illumina.basespace.File,%20long,%20long)

No Java API support for uploading files.

23andMe

For authentication 23andMe uses OAuth with bearer token. App developers must request an app key (client key and client secret in the 23andMe docs) which starts a multi-step process to generate a bearer token. This bearer token is then passed along in all the API calls.

The java library below uses the Scribe API https://github.com/fernandezpablo85/scribe-java to implement the authentication parts.

Support is provided for download of various types of data; for this API I think the only relevant calls will be to return the "genome" for a profile, which is actually a packed encoding of all SNPs into a single string. This could be converted to VCF on the fly and presented as a file to download.

https://api.23andme.com/docs/reference/#genomes

https://github.com/heuermh/personal-genome-client/blob/master/client/src/main/java/com/github/heuermh/personalgenome/client/PersonalGenomeClient.java

No support for uploading files.

1000 genomes
Download via FTP or Aspera
ftp://ftp-trace.ncbi.nih.gov/1000genomes/ftp/

Data types:
raw reads (fastq)
    /1000genomes/ftp/sequence.index => list of relative paths to fastq files

alignments (BAM)
    /1000genomes/ftp/alignments.index => llist of relative paths to bam files

old releases of index files are found in alignment_indices or sequence_indices

no authentication required

md5s are stored in index files - used by api to validate download?

1000 Genomes data are also available in AWS
http://www.1000genomes.org/using-1000-genomes-data-amazon-web-service-cloud

Link to Python version:
https://github.com/roryk/python-gcon
ping me with your github name and I'll add you as a committer.