Python

Jun 07, 2021

Contents

CHAPTER 1

Apollo API Library

Apollo is a Python library for interacting with Apollo. Arrow is its companion CLI tool.

> for the record ... arrow makes working with Apollo SOOO much easier — Nathan Dunn, Apollo Developer

1.1 Installation

```
pip install apollo
```

1.2 Examples

Example code to create a new organism and add yourself in the permission list:

```
from apollo import ApolloInstance
wa = ApolloInstance('https://fqdn/apollo', 'jane.doe@fqdn.edu', 'password')

orgs = wa.organisms.add_organism(
    "Yeast",
    "/path/to/jbrowse/data",
    genus='Saccharomyces',
    species='cerevisiae',
    public=False
)

# Give Apollo a second to process the uploaded organism.
time.sleep(1)
```

(continues on next page)

(continued from previous page)

```
# Then add yourself to permission list
data = wa.users.update_organism_permissions(
    "jane.doe@fqdn.edu",
    "Yeast",
    write=True,
    export=True,
    read=True,
)
```

If you have already created an Arrow config file (with command *arrow init*), you can also get an ApolloInstance without writing credentials explicitely:

```
from arrow.apollo import get_apollo_instance
wa = get_apollo_instance()
```

Or with the Arrow client:

```
$ arrow groups create_group university
{
    "publicGroup": false,
    "class": "org.bbop.apollo.UserGroup",
    "name": "university",
    "users": null,
    "id": 558319
}
# THEN
$ arrow users get_users | \
    jq '.[] | select(.username | contains("@tamu.edu")) | .username' | \
    xargs -n1 arrow users add_to_group university
# OR
$ arrow users get_users | \
    jq '.[] | select(.username | contains("@tamu.edu")) | .username' | \
    paste -s -d',' | \
    xargs arrow group update_membership 558319 --users
```

1.3 History

- **4.2.13**
  - Relax biopython requirements
- **4.2.12**
  - Do not filter out username from api responses
- **4.2.11**
  - Updated wrong version number
- **4.2.10**
  - Bugfix handling Shine-Dalgarno sequences (https://github.com/galaxy-genome-annotation/python-apollo/issues/48)
- **4.2.9**
  - Bugfix to update_organism when using suppress_output

- **4.2.8**
  - Added –suppress_output to update_organism
- **4.2.7**
  - Renamed –return_all option to –suppress_output
- **4.2.6**
  - Added –return_all option to add_organism and delete_organism methods
- **4.2.5**
  - Prevent from displaying login/password in the logs
- **4.2.4**
  - Remove unused dependency
- **4.2.3**
  - Fixed *load_gff3* to more accurately load transcripts including the CDS as well as handle non-coding types more accurately.
- **4.2.2**
  - Drastically speed up load_gff3
  - *load_gff3* now uses the Apollo *add_transcript* method if it is a gene or mRNA type
  - Added support for all of the current Apollo coding and non-coding types
  - Drop support for Python 2.7
- **4.2.1**
  - Fix getting groups by name
  - Add tests for group api
- **4.2**
  - Improve user update method
  - Add tests for user api
- **4.1**
  - Fix loading attributes from gff3
  - Better handling of genome sequence update, with or without the no_reload_sequences option
- **4.0.1**
  - Fix missing file in pypi package, no code change
- **4.0**
  - Added support for remote creation/update/deletion of organisms/tracks
  - Added support for adding GFF3 in the annotation track
  - Added tests
- **3.1**
  - Added user activate/inactivate
  - Added get_creator for user, group and organisms

  - Added omitEmptyOrganisms to get_users
  - Added support for group admins
  - Added support for bulk group creation/deletion
  - Repaired GFF3/Fasta downloading
- **3.0.4**
  - Fixed bug in deleteFeatures (Thanks @NeillGibson)
- **3.0.3**
  - findAllOrganisms works correctly, client side filtering no longer necessary.
- **3.0.2**
  - Patch a bug discovered in io.write, thanks Morgan!
- **3.0**
  - "Arrow" CLI utility
  - More pythonic API and many workarounds for Apollo bugs or oddities
  - Complete package restructure
  - Nearly all functions renamed
- **2.0**
  - Galaxy Functions
  - TTL Cache to work around Galaxy's behaviour
  - Status and Canned* Clients from @abretaud
- **1.0**
  - Initial release

1.4 Development

The content of docs and arrow directories is automatically generated from the code in the apollo directory. To regenerate it, install the latest version of the code, then run:

```
make rebuild
```

1.5 License

Available under the MIT License

1.6 Support

This material is based upon work supported by the National Science Foundation under Grant Number (Award 1565146)

CHAPTER 2

Commands

Arrow is a set of wrappers for Apollo's API. It builds a set of small, useful utilities for talking to Apollo servers. Each utility is implemented as a subcommand of `arrow`. This section of the documentation describes these commands.

2.1 annotations

This section is auto-generated from the help text for the arrow command `annotations`.

2.1.1 `add_attribute` command

Usage:

```
arrow annotations add_attribute [OPTIONS] FEATURE_ID ATTRIBUTE_KEY
```

Help

Add an attribute to a feature

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.2 `add_comment` command

Usage:

```
arrow annotations add_comment [OPTIONS] FEATURE_ID
```

Help

Set a feature's description

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--comments TEXT  Feature comments
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.3 `add_dbxref` command

Usage:

```
arrow annotations add_dbxref [OPTIONS] FEATURE_ID DB ACCESSION
```

Help

Add a dbxref to a feature

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.4 `add_feature` command

Usage:

```
arrow annotations add_feature [OPTIONS]
```

Help

Add a single feature

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--feature TEXT   Feature information
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.5 `add_features` command

Usage:

```
arrow annotations add_features [OPTIONS]
```

Help

Add a list of feature

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--features TEXT  Feature information
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.6 `add_transcript` command

Usage:

```
arrow annotations add_transcript [OPTIONS]
```

Help

Add a single transcript annotation

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--transcript TEXT   Transcript data
--suppress_history  Suppress the history of this operation
--suppress_events   Suppress instant update of the user interface
--organism TEXT     Organism Common Name
--sequence TEXT     Sequence Name
-h, --help          Show this message and exit.
```

2.1.7 `add_transcripts` command

Usage:

```
arrow annotations add_transcripts [OPTIONS]
```

Help

Add a list of transcript annotations

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--transcripts TEXT  Transcript data
--suppress_history  Suppress the history of this operation
--suppress_events   Suppress instant update of the user interface
--organism TEXT     Organism Common Name
--sequence TEXT     Sequence Name
-h, --help          Show this message and exit.
```

2.1.8 `delete_attribute` command

Usage:

```
arrow annotations delete_attribute [OPTIONS] FEATURE_ID ATTRIBUTE_KEY
```

Help

Delete an attribute from a feature

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.9 `delete_dbxref` command

Usage:

```
arrow annotations delete_dbxref [OPTIONS] FEATURE_ID DB ACCESSION
```

Help

Delete a dbxref from a feature

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.10 `delete_feature` command

Usage:

```
arrow annotations delete_feature [OPTIONS] FEATURE_ID
```

Help

Delete a feature

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.11 `delete_sequence_alteration` command

Usage:

```
arrow annotations delete_sequence_alteration [OPTIONS] FEATURE_ID
```

Help

[UNTESTED] Delete a specific feature alteration

Output

A list of sequence alterations(?)

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.12 `duplicate_transcript` command

Usage:

```
arrow annotations duplicate_transcript [OPTIONS] TRANSCRIPT_ID
```

Help

Duplicate a transcripte

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.13 `flip_strand` command

Usage:

```
arrow annotations flip_strand [OPTIONS] FEATURE_ID
```

Help

Flip the strand of a feature

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.14 `get_comments` command

Usage:

```
arrow annotations get_comments [OPTIONS] FEATURE_ID
```

Help

Get a feature's comments

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.15 `get_feature_sequence` command

Usage:

```
arrow annotations get_feature_sequence [OPTIONS] FEATURE_ID
```

Help

[CURRENTLY BROKEN] Get the sequence of a feature

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.16 `get_features` command

Usage:

```
arrow annotations get_features [OPTIONS]
```

Help

Get the features for an organism / sequence

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.17 `get_gff3` command

Usage:

```
arrow annotations get_gff3 [OPTIONS] FEATURE_ID
```

Help

Get the GFF3 associated with a feature

Output

GFF3 text content

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.18 `get_search_tools` command

Usage:

```
arrow annotations get_search_tools [OPTIONS]
```

Help

Get the search tools available

Output

dictionary containing the search tools and their metadata. For example:

```
{
    "sequence_search_tools": {
        "blat_prot": {
            "name": "Blat protein",
```

(continues on next page)

(continued from previous page)

```
            "search_class": "org.bbop.apollo.sequence.search.blat.
↪BlatCommandLineProteinToNucleotide",
            "params": "",
            "search_exe": "/usr/local/bin/blat"
        },
        "blat_nuc": {
            "name": "Blat nucleotide",
            "search_class": "org.bbop.apollo.sequence.search.blat.
↪BlatCommandLineNucleotideToNucleotide",
            "params": "",
            "search_exe": "/usr/local/bin/blat"
        }
    }
}
```

Options:

```
-h, --help  Show this message and exit.
```

2.1.19 `get_sequence_alterations` command

Usage:

```
arrow annotations get_sequence_alterations [OPTIONS]
```

Help

[UNTESTED] Get all of the sequence's alterations

Output

A list of sequence alterations(?)

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.20 `load_gff3` command

Usage:

```
arrow annotations load_gff3 [OPTIONS] ORGANISM GFF3
```

Help

Load a full GFF3 into annotation track

Output

Loading report

Options:

```
--source TEXT                 URL where the input dataset can be found.
--batch_size INTEGER          Size of batches before writing  [default: 1]
--test                        Run in dry run mode
--use_name                    Use the given name instead of generating one.
--disable_cds_recalculation   Disable CDS recalculation and instead use the one
                              provided

--timing                      Output loading performance metrics
-h, --help                    Show this message and exit.
```

2.1.21 `load_legacy_gff3` command

Usage:

```
arrow annotations load_legacy_gff3 [OPTIONS] ORGANISM GFF3
```

Help

Load a full GFF3 into annotation track (legacy version, kept for compatibility only)

Output

Loading report

Options:

```
--source TEXT  URL where the input dataset can be found.
-h, --help     Show this message and exit.
```

2.1.22 `merge_exons` command

Usage:

```
arrow annotations merge_exons [OPTIONS] EXON_A EXON_B
```

Help

Merge two exons

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.23 `set_boundaries` command

Usage:

```
arrow annotations set_boundaries [OPTIONS] FEATURE_ID START END
```

Help

Set the boundaries of a genomic feature

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.24 `set_description` command

Usage:

```
arrow annotations set_description [OPTIONS] FEATURE_ID DESCRIPTION
```

Help

Set a feature's description

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.25 `set_longest_orf` command

Usage:

```
arrow annotations set_longest_orf [OPTIONS] FEATURE_ID
```

Help

Automatically pick the longest ORF in a feature

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.26 `set_name` command

Usage:

```
arrow annotations set_name [OPTIONS] FEATURE_ID NAME
```

Help

Set a feature's name

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.27 `set_readthrough_stop_codon` command

Usage:

```
arrow annotations set_readthrough_stop_codon [OPTIONS] FEATURE_ID
```

Help

Set the feature to read through the first encountered stop codon

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.28 `set_sequence` command

Usage:

```
arrow annotations set_sequence [OPTIONS] ORGANISM SEQUENCE
```

Help

Set the sequence for subsequent requests. Mostly used in client scripts to avoid passing the sequence and organism on every function call.

Output

None

Options:

```
-h, --help  Show this message and exit.
```

2.1.29 `set_status` command

Usage:

```
arrow annotations set_status [OPTIONS] FEATURE_ID STATUS
```

Help

Set a feature's description

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.30 `set_symbol` command

Usage:

```
arrow annotations set_symbol [OPTIONS] FEATURE_ID SYMBOL
```

Help

Set a feature's description

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.31 `set_translation_end` command

Usage:

```
arrow annotations set_translation_end [OPTIONS] FEATURE_ID END
```

Help

Set a feature's end

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.32 `set_translation_start` command

Usage:

```
arrow annotations set_translation_start [OPTIONS] FEATURE_ID START
```

Help

Set the translation start of a feature

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.33 `update_attribute` command

Usage:

```
arrow annotations update_attribute [OPTIONS] FEATURE_ID ATTRIBUTE_KEY
```

Help

Delete an attribute from a feature

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.1.34 `update_dbxref` command

Usage:

```
arrow annotations update_dbxref [OPTIONS] FEATURE_ID OLD_DB OLD_ACCESSION
```

Help

Delete a dbxref from a feature

Output

A standard apollo feature dictionary ({"features": [{...}]})

Options:

```
--organism TEXT  Organism Common Name
--sequence TEXT  Sequence Name
-h, --help       Show this message and exit.
```

2.2 cannedcomments

This section is auto-generated from the help text for the arrow command `cannedcomments`.

2.2.1 `add_comment` command

Usage:

```
arrow cannedcomments add_comment [OPTIONS] COMMENT
```

Help

Add a canned comment

Output

A dictionnary containing canned comment description

Options:

```
--metadata TEXT  Optional metadata
-h, --help       Show this message and exit.
```

2.2.2 `delete_comment` command

Usage:

```
arrow cannedcomments delete_comment [OPTIONS] ID_NUMBER
```

Help

Update a canned comment

Output

an empty dictionary

Options:

```
-h, --help  Show this message and exit.
```

2.2.3 `get_comments` command

Usage:

```
arrow cannedcomments get_comments [OPTIONS]
```

Help

Get all canned comments available in this Apollo instance

Output

list of canned comment info dictionaries

Options:

```
-h, --help  Show this message and exit.
```

2.2.4 `show_comment` command

Usage:

```
arrow cannedcomments show_comment [OPTIONS] VALUE
```

Help

Get a specific canned comment

Output

A dictionnary containing canned comment description

Options:

```
-h, --help  Show this message and exit.
```

2.2.5 `update_comment` command

Usage:

```
arrow cannedcomments update_comment [OPTIONS] ID_NUMBER NEW_VALUE
```

Help

Update a canned comment

Output

an empty dictionary

Options:

```
--metadata TEXT  Optional metadata
-h, --help       Show this message and exit.
```

2.3 cannedkeys

This section is auto-generated from the help text for the arrow command `cannedkeys`.

2.3.1 `add_key` command

Usage:

```
arrow cannedkeys add_key [OPTIONS] KEY
```

Help

Add a canned key

Output

A dictionnary containing canned key description

Options:

```
--metadata TEXT  Optional metadata
-h, --help       Show this message and exit.
```

2.3.2 `delete_key` command

Usage:

```
arrow cannedkeys delete_key [OPTIONS] ID_NUMBER
```

Help

Update a canned key

Output

an empty dictionary

Options:

```
-h, --help  Show this message and exit.
```

2.3.3 `get_keys` command

Usage:

```
arrow cannedkeys get_keys [OPTIONS]
```

Help

Get all canned keys available in this Apollo instance

Output

list of canned key info dictionaries

Options:

```
-h, --help  Show this message and exit.
```

2.3.4 `show_key` command

Usage:

```
arrow cannedkeys show_key [OPTIONS] VALUE
```

Help

Get a specific canned key

Output

A dictionnary containing canned key description

Options:

```
-h, --help  Show this message and exit.
```

2.3.5 `update_key` command

Usage:

```
arrow cannedkeys update_key [OPTIONS] ID_NUMBER NEW_KEY
```

Help

Update a canned key

Output

an empty dictionary

Options:

```
--metadata TEXT  Optional metadata
-h, --help       Show this message and exit.
```

2.4 cannedvalues

This section is auto-generated from the help text for the arrow command `cannedvalues`.

2.4.1 `add_value` command

Usage:

```
arrow cannedvalues add_value [OPTIONS] VALUE
```

Help

Add a canned value

Output

A dictionnary containing canned value description

Options:

```
--metadata TEXT  Optional metadata
-h, --help       Show this message and exit.
```

2.4.2 `delete_value` command

Usage:

```
arrow cannedvalues delete_value [OPTIONS] ID_NUMBER
```

Help

Update a canned value

Output

an empty dictionary

Options:

```
-h, --help  Show this message and exit.
```

2.4.3 `get_values` command

Usage:

```
arrow cannedvalues get_values [OPTIONS]
```

Help

Get all canned values available in this Apollo instance

Output

list of canned value info dictionaries

Options:

```
-h, --help  Show this message and exit.
```

2.4.4 `show_value` command

Usage:

```
arrow cannedvalues show_value [OPTIONS] VALUE
```

Help

Get a specific canned value

Output

A dictionnary containing canned value description

Options:

```
-h, --help  Show this message and exit.
```

2.4.5 `update_value` command

Usage:

```
arrow cannedvalues update_value [OPTIONS] ID_NUMBER NEW_VALUE
```

Help

Update a canned value

Output

an empty dictionary

Options:

```
--metadata TEXT  Optional metadata
-h, --help       Show this message and exit.
```

2.5 groups

This section is auto-generated from the help text for the arrow command `groups`.

2.5.1 `create_group` command

Usage:

```
arrow groups create_group [OPTIONS] NAME
```

Help

Create a new group

Output

Group information dictionary

Options:

```
-h, --help  Show this message and exit.
```

2.5.2 `delete_group` command

Usage:

```
arrow groups delete_group [OPTIONS] GROUP
```

Help

Delete a group

Output

an empty dictionary

Options:

```
-h, --help  Show this message and exit.
```

2.5.3 `get_group_admin` command

Usage:

```
arrow groups get_group_admin [OPTIONS] GROUP
```

Help

Get the group's admins

Output

a list containing group admins

Options:

```
-h, --help  Show this message and exit.
```

2.5.4 `get_group_creator` command

Usage:

```
arrow groups get_group_creator [OPTIONS] GROUP
```

Help

Get the group's creator

Output

creator userId

Options:

```
-h, --help  Show this message and exit.
```

2.5.5 `get_groups` command

Usage:

```
arrow groups get_groups [OPTIONS]
```

Help

Get all the groups

Output

list of a dictionaries containing group information

Options:

```
--name TEXT  Only return group(s) with given name
-h, --help   Show this message and exit.
```

2.5.6 `get_organism_permissions` command

Usage:

```
arrow groups get_organism_permissions [OPTIONS] GROUP
```

Help

Get the group's organism permissions

Output

a list containing organism permissions (if any)

Options:

```
-h, --help  Show this message and exit.
```

2.5.7 `show_group` command

Usage:

```
arrow groups show_group [OPTIONS] GROUP_ID
```

Help

Get information about a group

Output

a dictionary containing group information

Options:

```
-h, --help  Show this message and exit.
```

2.5.8 `update_group` command

Usage:

```
arrow groups update_group [OPTIONS] GROUP_ID NEW_NAME
```

Help

Update the name of a group

Output

a dictionary containing group information

Options:

```
-h, --help  Show this message and exit.
```

2.5.9 `update_group_admin` command

Usage:

```
arrow groups update_group_admin [OPTIONS] GROUP_ID
```

Help

Update the group's admins

Output

dictionary of group information

Options:

```
--users TEXT  List of emails
-h, --help    Show this message and exit.
```

2.5.10 `update_membership` command

Usage:

```
arrow groups update_membership [OPTIONS]
```

Help

Update the group's membership

Output

dictionary of group information

Options:

```
--group_id INTEGER  Group ID Number
--users TEXT        List of emails
--memberships TEXT  Bulk memberships to update of the form: [ {groupId:
                    <groupId>,users: ["user1", "user2", "user3"]},
                    {groupId:<another-groupId>, users: ["user2", "user8"]}
                    (users and groupId will be ignored)

-h, --help          Show this message and exit.
```

2.5.11 `update_organism_permissions` command

Usage:

```
arrow groups update_organism_permissions [OPTIONS] GROUP ORGANISM_NAME
```

Help

Update the group's permissions on an organism

Output

list of group organism permissions

Options:

```
--administrate  Should the group have administrate privileges
--write         Should the group have write privileges
--read          Should the group have read privileges
--export        Should the group have export privileges
-h, --help      Show this message and exit.
```

2.6 io

This section is auto-generated from the help text for the arrow command `io`.

2.6.1 `download` command

Usage:

```
arrow io download [OPTIONS] UUID
```

Help

Download pre-prepared data by UUID

Output

The downloaded content

Options:

```
--output_format TEXT  Output format of the data, either "gzip" or "text"
                      [default: gzip]

-h, --help            Show this message and exit.
```

2.6.2 `write_downloadable` command

Usage:

```
arrow io write_downloadable [OPTIONS] ORGANISM
```

Help

Prepare a download for an organism

Output

a dictionary containing download information

Options:

```
--export_type TEXT    Export type. Choices: FASTA, GFF3, VCF  [default: FASTA]
--seq_type TEXT       Export selection. Choices: peptide, cds, cdna, genomic
                      [default: peptide]

--export_format TEXT  Export format, either gzip or text  [default: text]
--export_gff3_fasta   Export reference sequence when exporting GFF3
                      annotations.

--sequences TEXT      Names of references sequences to add (default is all)
--region TEXT         Region to export in form sequence:min..max e.g.,
                      chr3:1001..1034

-h, --help            Show this message and exit.
```

2.6.3 `write_text` command

Usage:

```
arrow io write_text [OPTIONS] ORGANISM
```

Help

[DEPRECATED, use write_downloadable] Download or prepare a download for an organism

Output

the exported data

Options:

```
--export_type TEXT    Export type. Choices: FASTA, GFF3, VCF  [default: FASTA]
--seq_type TEXT       Export selection. Choices: peptide, cds, cdna, genomic
                      [default: peptide]

--export_format TEXT  Export format, either gzip or text  [default: text]
--export_gff3_fasta   Export reference sequence when exporting GFF3
                      annotations.

--sequences TEXT      Names of references sequences to add (default is all)
--region TEXT         Region to export in form sequence:min..max e.g.,
                      chr3:1001..1034

-h, --help            Show this message and exit.
```

2.7 metrics

This section is auto-generated from the help text for the arrow command `metrics`.

2.7.1 `get_metrics` command

Usage:

```
arrow metrics get_metrics [OPTIONS]
```

Help

Get all server metrics

Output

A dictionary with all of the server timing / metrics

Options:

```
-h, --help  Show this message and exit.
```

2.8 organisms

This section is auto-generated from the help text for the arrow command `organisms`.

2.8.1 `add_organism` command

Usage:

```
arrow organisms add_organism [OPTIONS] COMMON_NAME DIRECTORY
```

Help

Add an organism

Output

a dictionary with information about the new organism

Options:

```
--blatdb TEXT       Server-side path to 2bit index of the genome for Blat
--genus TEXT        Genus
--species TEXT      Species
--public            Should the organism be public or not
--metadata TEXT     JSON formatted arbitrary metadata
--suppress_output   Suppress output of all organisms (true / false) (default
                    false)

-h, --help          Show this message and exit.
```

2.8.2 `delete_features` command

Usage:

```
arrow organisms delete_features [OPTIONS] ORGANISM_ID
```

Help

Remove features of an organism

Output

an empty dictionary

Options:

```
-h, --help  Show this message and exit.
```

2.8.3 `delete_organism` command

Usage:

```
arrow organisms delete_organism [OPTIONS] ORGANISM_ID
```

Help

Delete an organism

Output

A list of all remaining organisms

Options:

```
--suppress_output   Suppress return of all organisms (true / false) (default
                    false)

-h, --help          Show this message and exit.
```

2.8.4 `get_organism_creator` command

Usage:

```
arrow organisms get_organism_creator [OPTIONS] ORGANISM_ID
```

Help

Get the creator of an organism

Output

a dictionary containing user information

Options:

```
-h, --help  Show this message and exit.
```

2.8.5 `get_organisms` command

Usage:

```
arrow organisms get_organisms [OPTIONS]
```

Help

Get all organisms

Output

Organism information

Options:

```
--common_name TEXT  Optionally filter on common name
-h, --help          Show this message and exit.
```

2.8.6 `get_sequences` command

Usage:

```
arrow organisms get_sequences [OPTIONS] ORGANISM_ID
```

Help

Get the sequences for an organism

Output

The set of sequences associated with an organism

Options:

```
-h, --help  Show this message and exit.
```

2.8.7 `show_organism` command

Usage:

```
arrow organisms show_organism [OPTIONS] COMMON_NAME
```

Help

Get information about a specific organism.

Output

a dictionary containing the organism's information

Options:

```
-h, --help  Show this message and exit.
```

2.8.8 `update_metadata` command

Usage:

```
arrow organisms update_metadata [OPTIONS] ORGANISM_ID METADATA
```

Help

Update the metadata for an existing organism.

Output

An empty, useless dictionary

Options:

```
-h, --help  Show this message and exit.
```

2.8.9 `update_organism` command

Usage:

```
arrow organisms update_organism [OPTIONS] ORGANISM_ID COMMON_NAME
```

Help

Update an organism

Output

a dictionary with information about the updated organism

Options:

```
--blatdb TEXT           Server-side Blat directory for the organism
--species TEXT          Species
--genus TEXT            Genus
--public                User's email
--no_reload_sequences   Set this if you don't want Apollo to reload genome
                        sequences (no change in genome sequence)
```

(continues on next page)

(continued from previous page)

```
--suppress_output       Suppress output of all organisms (true / false)
                        (default false)

-h, --help              Show this message and exit.
```

2.9 remote

This section is auto-generated from the help text for the arrow command `remote`.

2.9.1 `add_organism` command

Usage:

```
arrow remote add_organism [OPTIONS] COMMON_NAME ORGANISM_DATA
```

Help

Add an organism using the remote organism API.

Output

a dictionary with information about the new organism

Options:

```
--blatdb FILENAME               Path to 2bit index of the genome for Blat
                                (Blat 2bit data can also be in organism_data
                                in directory 'searchDatabaseData')

--genus TEXT                    Genus
--species TEXT                  Species
--public                        should the organism be public
--non_default_translation_table INTEGER
                                The translation table number for the organism
                                (if different than that of the server's
                                default)

--metadata TEXT                 JSON formatted arbitrary metadata
-h, --help                      Show this message and exit.
```

2.9.2 `add_track` command

Usage:

```
arrow remote add_track [OPTIONS] ORGANISM_ID TRACK_DATA TRACK_CONFIG
```

Help

Adds a tarball containing track data to an existing organism.

Output

a dictionary with information about all tracks on the organism

Options:

```
-h, --help  Show this message and exit.
```

2.9.3 `delete_organism` command

Usage:

```
arrow remote delete_organism [OPTIONS] ORGANISM_ID
```

Help

Remove an organism completely.

Output

a dictionary with information about the deleted organism

Options:

```
-h, --help  Show this message and exit.
```

2.9.4 `delete_track` command

Usage:

```
arrow remote delete_track [OPTIONS] ORGANISM_ID TRACK_LABEL
```

Help

Remove a track from an organism

Output

a dictionary with information about the deleted track

Options:

```
-h, --help  Show this message and exit.
```

2.9.5 `update_organism` command

Usage:

```
arrow remote update_organism [OPTIONS] ORGANISM_ID ORGANISM_DATA
```

Help

Update an organism using the remote organism API.

Output

a dictionary with information about the updated organism

Options:

```
--blatdb FILENAME       Path to 2bit index of the genome for Blat (Blat 2bit
                        data can also be in organism_data in directory
                        'searchDatabaseData')

--common_name TEXT      Organism common name
--genus TEXT            Genus
--species TEXT          Species
--public                User's email
--metadata TEXT         JSON formatted arbitrary metadata
--no_reload_sequences   Set this if you don't want Apollo to reload genome
                        sequences (no change in genome sequence)

-h, --help              Show this message and exit.
```

2.9.6 `update_track` command

Usage:

```
arrow remote update_track [OPTIONS] ORGANISM_ID TRACK_CONFIG
```

Help

Update the configuration of a track that has already been added to the organism. Will not update data for the track.

Output

a dictionary with information about all tracks on the organism

Options:

```
-h, --help  Show this message and exit.
```

2.10 status

This section is auto-generated from the help text for the arrow command `status`.

2.10.1 `add_status` command

Usage:

```
arrow status add_status [OPTIONS] STATUS
```

Help

Add a status value

Output

A dictionnary containing status description

Options:

```
-h, --help  Show this message and exit.
```

2.10.2 `delete_status` command

Usage:

```
arrow status delete_status [OPTIONS] ID_NUMBER
```

Help

Delete a status

Output

an empty dictionary

Options:

```
-h, --help  Show this message and exit.
```

2.10.3 `get_statuses` command

Usage:

```
arrow status get_statuses [OPTIONS]
```

Help

Get all statuses available in this Apollo instance

Output

list of status info dictionaries

Options:

```
-h, --help  Show this message and exit.
```

2.10.4 `show_status` command

Usage:

```
arrow status show_status [OPTIONS] STATUS
```

Help

Get a specific status

Output

A dictionnary containing status description

Options:

```
-h, --help  Show this message and exit.
```

2.10.5 `update_status` command

Usage:

```
arrow status update_status [OPTIONS] ID_NUMBER NEW_VALUE
```

Help

Update a status name

Output

an empty dictionary

Options:

```
-h, --help  Show this message and exit.
```

2.11 users

This section is auto-generated from the help text for the arrow command `users`.

2.11.1 `activate_user` command

Usage:

```
arrow users activate_user [OPTIONS] USER
```

Help

Activate a user

Output

an empty dictionary

Options:

```
-h, --help  Show this message and exit.
```

2.11.2 `add_to_group` command

Usage:

```
arrow users add_to_group [OPTIONS] GROUP USER
```

Help

Add a user to a group

Output

an empty dictionary

Options:

```
-h, --help  Show this message and exit.
```

2.11.3 `create_user` command

Usage:

```
arrow users create_user [OPTIONS] EMAIL FIRST_NAME LAST_NAME PASSWORD
```

Help

Create a new user

Output

an empty dictionary

Options:

```
--role TEXT      User's default role, one of "admin" or "user"  [default:
                 user]

--metadata TEXT  User metadata
-h, --help       Show this message and exit.
```

2.11.4 `delete_user` command

Usage:

```
arrow users delete_user [OPTIONS] USER
```

Help

Delete a user

Output

an empty dictionary

Options:

```
-h, --help  Show this message and exit.
```

2.11.5 `get_organism_permissions` command

Usage:

```
arrow users get_organism_permissions [OPTIONS] USER
```

Help

Display a user's organism permissions

Output

a dictionary containing user's organism permissions

Options:

```
-h, --help  Show this message and exit.
```

2.11.6 `get_user_creator` command

Usage:

```
arrow users get_user_creator [OPTIONS] USER
```

Help

Get the creator of a user

Output

a dictionary containing user information

Options:

```
-h, --help  Show this message and exit.
```

2.11.7 `get_users` command

Usage:

```
arrow users get_users [OPTIONS]
```

Help

Get all users known to this Apollo instance

Output

list of user info dictionaries

Options:

```
--omit_empty_organisms  Will omit users having no access to any organism
-h, --help              Show this message and exit.
```

2.11.8 `inactivate_user` command

Usage:

```
arrow users inactivate_user [OPTIONS] USER
```

Help

Activate a user

Output

an empty dictionary

Options:

```
-h, --help  Show this message and exit.
```

2.11.9 `remove_from_group` command

Usage:

```
arrow users remove_from_group [OPTIONS] GROUP USER
```

Help

Remove a user from a group

Output

an empty dictionary

Options:

```
-h, --help  Show this message and exit.
```

2.11.10 `show_user` command

Usage:

```
arrow users show_user [OPTIONS] USER
```

Help

Get a specific user

Output

a dictionary containing user information

Options:

```
-h, --help  Show this message and exit.
```

2.11.11 `update_organism_permissions` command

Usage:

```
arrow users update_organism_permissions [OPTIONS] USER ORGANISM
```

Help

Update the permissions of a user on a specified organism

Output

a dictionary containing user's organism permissions

Options:

```
--administrate  Grants administrative privileges
--write         Grants write privileges
--export        Grants export privileges
--read          Grants read privileges
-h, --help      Show this message and exit.
```

2.11.12 `update_user` command

Usage:

```
arrow users update_user [OPTIONS] EMAIL FIRST_NAME LAST_NAME
```

Help

Update an existing user

Output

a dictionary containing user information

Options:

```
--password TEXT   User's password (omit to keep untouched)
--metadata TEXT   User metadata
--new_email TEXT  User's new email (if you want to change it)
-h, --help        Show this message and exit.
```

CHAPTER 3

apollo package

3.1 Subpackages

3.1.1 apollo.annotations package

Module contents

Contains possible interactions with the Apollo's Annotations

class apollo.annotations.**AnnotationsClient**(*webapolloinstance*, ***requestArgs*)
Bases: *apollo.client.Client*

CLIENT_BASE = '/annotationEditor/'

add_attribute(*feature_id*, *attribute_key*, *attribute_value*, *organism=None*, *sequence=None*)
Add an attribute to a feature

Parameters

- **feature_id** (*str*) – Feature UUID
- **attribute_key** (*str*) – Attribute Key
- **attribute_value** (*str*) – Attribute Value
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

This seems to show two attributes being added, but it behaves like those two are one.

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

add_comment(*feature_id*, *comments=[]*, *organism=None*, *sequence=None*)
Set a feature's description

Parameters

- **feature_id** (*str*) – Feature UUID
- **comments** (*list*) – Feature comments
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

add_dbxref(*feature_id*, *db*, *accession*, *organism=None*, *sequence=None*)
Add a dbxref to a feature

Parameters

- **feature_id** (*str*) – Feature UUID
- **db** (*str*) – DB Name (e.g. PMID)
- **accession** (*str*) – Accession Value
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

This seems to show two attributes being added, but it behaves like those two are one.

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

add_feature(*feature={}*, *organism=None*, *sequence=None*)
Add a single feature

Parameters

- **feature** (*dict*) – Feature information
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

add_features(*features=[]*, *organism=None*, *sequence=None*)
Add a list of feature

Parameters

- **features** (*list*) – Feature information
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

add_transcript(*transcript={}*, *suppress_history=False*, *suppress_events=False*, *organism=None*, *sequence=None*)
Add a single transcript annotation

Parameters

- **transcript** (*dict*) – Transcript data

- **suppress_history** (*bool*) – Suppress the history of this operation
- **suppress_events** (*bool*) – Suppress instant update of the user interface
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

add_transcripts(*transcripts=[]*, *suppress_history=False*, *suppress_events=False*, *organism=None*, *sequence=None*)

Add a list of transcript annotations

Parameters

- **transcripts** (*list*) – Transcript data
- **suppress_history** (*bool*) – Suppress the history of this operation
- **suppress_events** (*bool*) – Suppress instant update of the user interface
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

delete_attribute(*feature_id*, *attribute_key*, *attribute_value*, *organism=None*, *sequence=None*)

Delete an attribute from a feature

Parameters

- **feature_id** (*str*) – Feature UUID
- **attribute_key** (*str*) – Attribute Key
- **attribute_value** (*str*) – Attribute Value
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

delete_dbxref(*feature_id*, *db*, *accession*, *organism=None*, *sequence=None*)

Delete a dbxref from a feature

Parameters

- **feature_id** (*str*) – Feature UUID
- **db** (*str*) – DB Name (e.g. PMID)
- **accession** (*str*) – Accession Value
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

delete_feature(*feature_id*, *organism=None*, *sequence=None*)
Delete a feature

Parameters

- **feature_id** (*str*) – Feature UUID
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

delete_sequence_alteration(*feature_id*, *organism=None*, *sequence=None*)
[UNTESTED] Delete a specific feature alteration

Parameters

- **feature_id** (*str*) – Feature UUID
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type list

Returns A list of sequence alterations(?)

duplicate_transcript(*transcript_id*, *organism=None*, *sequence=None*)
Duplicate a transcripte

Parameters

- **transcript_id** (*str*) – Transcript UUID
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

flip_strand(*feature_id*, *organism=None*, *sequence=None*)
Flip the strand of a feature

Parameters

- **feature_id** (*str*) – Feature UUID
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

get_comments(*feature_id*, *organism=None*, *sequence=None*)
Get a feature's comments

Parameters

- **feature_id** (*str*) – Feature UUID
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

get_feature_sequence (*feature_id*, *organism=None*, *sequence=None*)
[CURRENTLY BROKEN] Get the sequence of a feature

Parameters

- **feature_id** (*str*) – Feature UUID
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

get_features (*organism=None*, *sequence=None*)
Get the features for an organism / sequence

Parameters

- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

get_gff3 (*feature_id*, *organism=None*, *sequence=None*)
Get the GFF3 associated with a feature

Parameters

- **feature_id** (*str*) – Feature UUID
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type str

Returns GFF3 text content

get_search_tools ()
Get the search tools available

Return type dict

Returns

dictionary containing the search tools and their metadata. For example:

```
{
    "sequence_search_tools": {
        "blat_prot": {
            "name": "Blat protein",
            "search_class": "org.bbop.apollo.sequence.search.blat.
↪BlatCommandLineProteinToNucleotide",
            "params": "",
            "search_exe": "/usr/local/bin/blat"
        },
        "blat_nuc": {
            "name": "Blat nucleotide",
```

(continues on next page)

(continued from previous page)

```
            "search_class": "org.bbop.apollo.sequence.search.blat.
↪BlatCommandLineNucleotideToNucleotide",
            "params": "",
            "search_exe": "/usr/local/bin/blat"
        }
    }
}
```

get_sequence_alterations(*organism=None*, *sequence=None*)
[UNTESTED] Get all of the sequence's alterations

Parameters

- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type list

Returns A list of sequence alterations(?)

load_gff3(*organism*, *gff3*, *source=None*, *batch_size=1*, *test=False*, *use_name=False*, *disable_cds_recalculation=False*, *timing=False*)
Load a full GFF3 into annotation track

Parameters

- **organism** (*str*) – Organism Common Name
- **gff3** (*str*) – GFF3 to load
- **source** (*str*) – URL where the input dataset can be found.
- **batch_size** (*int*) – Size of batches before writing
- **test** (*bool*) – Run in dry run mode
- **use_name** (*bool*) – Use the given name instead of generating one.
- **disable_cds_recalculation** (*bool*) – Disable CDS recalculation and instead use the one provided
- **timing** (*bool*) – Output loading performance metrics

Return type str

Returns Loading report

load_legacy_gff3(*organism*, *gff3*, *source=None*)
Load a full GFF3 into annotation track (legacy version, kept for compatibility only)

Parameters

- **organism** (*str*) – Organism Common Name
- **gff3** (*str*) – GFF3 to load
- **source** (*str*) – URL where the input dataset can be found.

Return type str

Returns Loading report

merge_exons(*exon_a*, *exon_b*, *organism=None*, *sequence=None*)
Merge two exons

Parameters

- **exon_a** (*str*) – Feature UUID
- **exon_b** (*str*) – Feature UUID
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

set_boundaries (*feature_id*, *start*, *end*, *organism=None*, *sequence=None*)
Set the boundaries of a genomic feature

Parameters

- **feature_id** (*str*) – Feature UUID
- **start** (*int*) – Feature start
- **end** (*int*) – Feature end
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

set_description (*feature_id*, *description*, *organism=None*, *sequence=None*)
Set a feature's description

Parameters

- **feature_id** (*str*) – Feature UUID
- **description** (*str*) – Feature description
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

set_longest_orf (*feature_id*, *organism=None*, *sequence=None*)
Automatically pick the longest ORF in a feature

Parameters

- **feature_id** (*str*) – Feature UUID
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

set_name (*feature_id*, *name*, *organism=None*, *sequence=None*)
Set a feature's name

Parameters

- **feature_id** (*str*) – Feature UUID
- **name** (*str*) – Feature name
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

set_readthrough_stop_codon (*feature_id*, *organism=None*, *sequence=None*)
Set the feature to read through the first encountered stop codon

Parameters

- **feature_id** (*str*) – Feature UUID
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

set_sequence (*organism*, *sequence*)
Set the sequence for subsequent requests. Mostly used in client scripts to avoid passing the sequence and organism on every function call.

Parameters

- **organism** (*str*) – Organism Name
- **sequence** (*str*) – Sequence Name

Return type None

Returns None

set_status (*feature_id*, *status*, *organism=None*, *sequence=None*)
Set a feature's description

Parameters

- **feature_id** (*str*) – Feature UUID
- **status** (*str*) – Feature status
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

set_symbol (*feature_id*, *symbol*, *organism=None*, *sequence=None*)
Set a feature's description

Parameters

- **feature_id** (*str*) – Feature UUID
- **symbol** (*str*) – Feature symbol
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

set_translation_end (*feature_id*, *end*, *organism=None*, *sequence=None*)
Set a feature's end

Parameters

- **feature_id** (*str*) – Feature UUID
- **end** (*int*) – Feature end
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

set_translation_start (*feature_id*, *start*, *organism=None*, *sequence=None*)
Set the translation start of a feature

Parameters

- **feature_id** (*str*) – Feature UUID
- **start** (*int*) – Feature start
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

update_attribute (*feature_id*, *attribute_key*, *old_value*, *new_value*, *organism=None*, *sequence=None*)
Delete an attribute from a feature

Parameters

- **feature_id** (*str*) – Feature UUID
- **attribute_key** (*str*) – Attribute Key
- **old_value** (*str*) – Old attribute value
- **new_value** (*str*) – New attribute value
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

update_dbxref (*feature_id*, *old_db*, *old_accession*, *new_db*, *new_accession*, *organism=None*, *sequence=None*)
Delete a dbxref from a feature

Parameters

- **feature_id** (*str*) – Feature UUID
- **old_db** (*str*) – Old DB Name (e.g. PMID)

- **old_accession** (*str*) – Old accession Value
- **new_db** (*str*) – New DB Name (e.g. PMID)
- **new_accession** (*str*) – New accession Value
- **organism** (*str*) – Organism Common Name
- **sequence** (*str*) – Sequence Name

Return type dict

Returns A standard apollo feature dictionary ({"features": [{...}]})

class apollo.annotations.**FeatureType**

Bases: enum.Enum

An enumeration.

FEATURE = 1

TRANSCRIPT = 2

3.1.2 apollo.cannedcomments package

Module contents

Contains possible interactions with the Apollo Canned Comments Module

class apollo.cannedcomments.**CannedCommentsClient**(*webapolloinstance*, ***requestArgs*)

Bases: *apollo.client.Client*

CLIENT_BASE = '/cannedComment/'

add_comment(*comment*, *metadata=''*)

Add a canned comment

Parameters

- **comment** (*str*) – New canned comment
- **metadata** (*str*) – Optional metadata

Return type dict

Returns A dictionnary containing canned comment description

delete_comment(*id_number*)

Update a canned comment

Parameters **id_number** (*int*) – canned comment ID number

Return type dict

Returns an empty dictionary

get_comments()

Get all canned comments available in this Apollo instance

Return type list of dicts

Returns list of canned comment info dictionaries

show_comment(*value*)

Get a specific canned comment

Parameters **value** (*str*) – Canned comment to show

Return type dict

Returns A dictionnary containing canned comment description

update_comment(*id_number*, *new_value*, *metadata=None*)
Update a canned comment

Parameters

- **id_number** (*int*) – canned comment ID number
- **new_value** (*str*) – New canned comment value
- **metadata** (*str*) – Optional metadata

Return type dict

Returns an empty dictionary

3.1.3 apollo.cannedkeys package

Module contents

Contains possible interactions with the Apollo Canned Keys Module

class apollo.cannedkeys.**CannedKeysClient**(*webapolloinstance*, ***requestArgs*)
Bases: *apollo.client.Client*

CLIENT_BASE = '/cannedKey/'

add_key(*key*, *metadata=''*)
Add a canned key

Parameters

- **key** (*str*) – New canned key
- **metadata** (*str*) – Optional metadata

Return type dict

Returns A dictionnary containing canned key description

delete_key(*id_number*)
Update a canned key

Parameters **id_number** (*int*) – canned key ID number

Return type dict

Returns an empty dictionary

get_keys()
Get all canned keys available in this Apollo instance

Return type list of dicts

Returns list of canned key info dictionaries

show_key(*value*)
Get a specific canned key

Parameters **value** (*str*) – Canned key to show

Return type dict

Returns A dictionnary containing canned key description

update_key(*id_number*, *new_key*, *metadata=None*)
Update a canned key

Parameters

- **id_number** (*int*) – canned key ID number
- **new_key** (*str*) – New canned key value
- **metadata** (*str*) – Optional metadata

Return type dict

Returns an empty dictionary

3.1.4 apollo.cannedvalues package

Module contents

Contains possible interactions with the Apollo Canned Values Module

class apollo.cannedvalues.**CannedValuesClient**(*webapolloinstance*, ***requestArgs*)
Bases: *apollo.client.Client*

CLIENT_BASE = '/cannedValue/'

add_value(*value*, *metadata=''*)
Add a canned value

Parameters

- **value** (*str*) – New canned value
- **metadata** (*str*) – Optional metadata

Return type dict

Returns A dictionnary containing canned value description

delete_value(*id_number*)
Update a canned value

Parameters **id_number** (*int*) – canned value ID number

Return type dict

Returns an empty dictionary

get_values()
Get all canned values available in this Apollo instance

Return type list of dicts

Returns list of canned value info dictionaries

show_value(*value*)
Get a specific canned value

Parameters **value** (*str*) – Canned value to show

Return type dict

Returns A dictionnary containing canned value description

update_value(*id_number*, *new_value*, *metadata=None*)
Update a canned value

Parameters

- **id_number** (*int*) – canned value ID number
- **new_value** (*str*) – New canned value value
- **metadata** (*str*) – Optional metadata

Return type dict

Returns an empty dictionary

3.1.5 apollo.groups package

Module contents

Contains possible interactions with the Apollo Groups

class apollo.groups.**GroupsClient**(*webapolloinstance*, ***requestArgs*)
Bases: *apollo.client.Client*

CLIENT_BASE = '/group/'

create_group(*name*)
Create a new group

Parameters **name** (*str*) – Group name (or a list of groups to create)

Return type dict

Returns Group information dictionary

delete_group(*group*)
Delete a group

Parameters **group** (*str*) – Group name (or a list of groups to delete)

Return type dict

Returns an empty dictionary

get_group_admin(*group*)
Get the group's admins

Parameters **group** (*str*) – group name

Return type list

Returns a list containing group admins

get_group_creator(*group*)
Get the group's creator

Parameters **group** (*str*) – group name

Return type list

Returns creator userId

get_groups(*name=None*)
Get all the groups

Parameters **name** (*str*) – Only return group(s) with given name

Return type list of dicts

Returns list of a dictionaries containing group information

get_organism_permissions(*group*)
Get the group's organism permissions

Parameters **group** (*str*) – group name

Return type list

Returns a list containing organism permissions (if any)

show_group(*group_id*)
Get information about a group

Parameters **group_id** (*int*) – Group ID Number

Return type dict

Returns a dictionary containing group information

update_group(*group_id*, *new_name*)
Update the name of a group

Parameters

- **group_id** (*int*) – group ID number
- **new_name** (*str*) – New name for the group

Return type dict

Returns a dictionary containing group information

update_group_admin(*group_id*, *users=[]*)
Update the group's admins

Parameters

- **group_id** (*int*) – Group ID Number
- **users** (*list of str*) – List of emails

Return type dict

Returns dictionary of group information

update_membership(*group_id=None*, *users=[]*, *memberships=[]*)
Update the group's membership

Parameters

- **group_id** (*int*) – Group ID Number
- **users** (*list of str*) – List of emails
- **memberships** (*list*) – Bulk memberships to update of the form: [{groupId: <groupId>,users: ["user1", "user2", "user3"]}, {groupId:<another-groupId>, users: ["user2", "user8"]} (users and groupId will be ignored)

Return type dict

Returns dictionary of group information

update_organism_permissions(*group*, *organism_name*, *administrate=False*, *write=False*, *read=False*, *export=False*)

Update the group's permissions on an organism

Parameters

- **group** (*str*) – group name
- **organism_name** (*str*) – Organism name
- **administrate** (*bool*) – Should the group have administrate privileges
- **read** (*bool*) – Should the group have read privileges
- **write** (*bool*) – Should the group have write privileges
- **export** (*bool*) – Should the group have export privileges

Return type list

Returns list of group organism permissions

3.1.6 apollo.io package

Module contents

Contains possible interactions with the Apollo IO Module

class apollo.io.**IOClient**(*webapolloinstance*, ***requestArgs*)

Bases: *apollo.client.Client*

CLIENT_BASE = '/IOService/'

download(*uuid*, *output_format='gzip'*)

Download pre-prepared data by UUID

Parameters

- **uuid** (*str*) – Data UUID
- **output_format** (*str*) – Output format of the data, either "gzip" or "text"

Return type str

Returns The downloaded content

write_downloadable(*organism*, *export_type='FASTA'*, *seq_type='peptide'*, *export_format='text'*, *export_gff3_fasta=False*, *sequences=[]*, *region=None*)

Prepare a download for an organism

Parameters

- **organism** (*str*) – organism common name
- **sequences** (*str*) – Names of references sequences to add (default is all)
- **export_type** (*str*) – Export type. Choices: FASTA, GFF3, VCF
- **seq_type** (*str*) – Export selection. Choices: peptide, cds, cdna, genomic
- **export_format** (*str*) – Export format, either gzip or text
- **export_gff3_fasta** (*bool*) – Export reference sequence when exporting GFF3 annotations.
- **region** (*str*) – Region to export in form sequence:min..max e.g., chr3:1001..1034

Return type dict

Returns a dictionary containing download information

write_text(*organism*, *export_type='FASTA'*, *seq_type='peptide'*, *export_format='text'*, *export_gff3_fasta=False*, *sequences=[]*, *region=None*)
[DEPRECATED, use write_downloadable] Download or prepare a download for an organism

Parameters

- **organism** (*str*) – organism common name
- **sequences** (*str*) – Names of references sequences to add (default is all)
- **export_type** (*str*) – Export type. Choices: FASTA, GFF3, VCF
- **seq_type** (*str*) – Export selection. Choices: peptide, cds, cdna, genomic
- **export_format** (*str*) – Export format, either gzip or text
- **export_gff3_fasta** (*bool*) – Export reference sequence when exporting GFF3 annotations.
- **region** (*str*) – Region to export in form sequence:min..max e.g., chr3:1001..1034

Return type str

Returns the exported data

3.1.7 apollo.metrics package

Module contents

Contains possible interactions with the Apollo Metrics

class apollo.metrics.**MetricsClient**(*webapolloinstance*, ***requestArgs*)
Bases: *apollo.client.Client*

CLIENT_BASE = '/metrics/'

get_metrics()
Get all server metrics

Return type dict

Returns A dictionary with all of the server timing / metrics

3.1.8 apollo.organisms package

Module contents

Contains possible interactions with the Apollo Organisms Module

class apollo.organisms.**OrganismsClient**(*webapolloinstance*, ***requestArgs*)
Bases: *apollo.client.Client*

CLIENT_BASE = '/organism/'

add_organism(*common_name*, *directory*, *blatdb=None*, *genus=None*, *species=None*, *public=False*, *metadata=None*, *suppress_output=False*)
Add an organism

Parameters

- **common_name** (*str*) – Organism common name
- **directory** (*str*) – Server-side directory
- **blatdb** (*str*) – Server-side path to 2bit index of the genome for Blat
- **genus** (*str*) – Genus
- **species** (*str*) – Species
- **public** (*bool*) – Should the organism be public or not
- **metadata** (*str*) – JSON formatted arbitrary metadata
- **suppress_output** (*bool*) – Suppress output of all organisms (true / false) (default false)

Return type dict

Returns a dictionary with information about the new organism

delete_features(*organism_id*)
Remove features of an organism

Parameters **organism_id** (*str*) – Organism ID Number

Return type dict

Returns an empty dictionary

delete_organism(*organism_id*, *suppress_output=False*)
Delete an organism

Parameters

- **organism_id** (*str*) – Organism ID Number
- **suppress_output** (*bool*) – Suppress return of all organisms (true / false) (default false)

Return type list

Returns A list of all remaining organisms

get_organism_creator(*organism_id*)
Get the creator of an organism

Parameters **organism_id** (*str*) – Organism ID Number

Return type dict

Returns a dictionary containing user information

get_organisms(*common_name=None*)
Get all organisms

Parameters **common_name** (*str*) – Optionally filter on common name

Return type list

Returns Organism information

get_sequences(*organism_id*)
Get the sequences for an organism

Parameters **organism_id** (*str*) – Organism ID Number

Return type list of dict

Returns The set of sequences associated with an organism

show_organism(*common_name*)
Get information about a specific organism.

Parameters **common_name** (*str*) – Organism Common Name

Return type dict

Returns a dictionary containing the organism's information

update_metadata(*organism_id*, *metadata*)
Update the metadata for an existing organism.

Parameters

- **organism_id** (*str*) – Organism ID Number
- **metadata** (*str*) – Organism metadata. (Recommendation: use a structured format like JSON)

Return type dict

Returns An empty, useless dictionary

update_organism(*organism_id*, *common_name*, *directory*, *blatdb=None*, *species=None*, *genus=None*, *public=False*, *no_reload_sequences=False*, *suppress_output=False*)
Update an organism

Parameters

- **organism_id** (*str*) – Organism ID Number
- **common_name** (*str*) – Organism common name
- **directory** (*str*) – Server-side directory
- **blatdb** (*str*) – Server-side Blat directory for the organism
- **genus** (*str*) – Genus
- **species** (*str*) – Species
- **public** (*bool*) – User's email
- **no_reload_sequences** (*bool*) – Set this if you don't want Apollo to reload genome sequences (no change in genome sequence)
- **suppress_output** (*bool*) – Suppress output of all organisms (true / false) (default false)

Return type dict

Returns a dictionary with information about the updated organism

3.1.9 apollo.remote package

Module contents

Contains possible interactions with the Apollo Organisms Module

class apollo.remote.**RemoteClient**(*webapolloinstance*, ***requestArgs*)
Bases: *apollo.client.Client*

CLIENT_BASE = '/organism/'

add_organism(*common_name*, *organism_data*, *blatdb=None*, *genus=None*, *species=None*, *public=None*, *non_default_translation_table=None*, *metadata=None*)

Add an organism using the remote organism API.

The recommended structure for the genome data tarball is as follows:

```
names/
names/root.json
searchDatabaseData/blat_db.2bit
seq/
seq/fba/
seq/fba/da8/
seq/fba/da8/f3/
seq/fba/da8/f3/Mijalis-0.txt
seq/fba/da8/f3/Mijalis-1.txt
seq/fba/da8/f3/Mijalis-2.txt
seq/fba/da8/f3/Mijalis-3.txt
seq/fba/da8/f3/Mijalis-4.txt
seq/refSeqs.json
tracks/
trackList.json
tracks.conf
```

The genome name / hashed directories below the seq folder will obviously be specific to your organism.

Parameters

- **common_name** (*str*) – Organism common name
- **organism_data** (*file*) – .tar.gz or .zip archive containing the data directory.
- **blatdb** (*file*) – Path to 2bit index of the genome for Blat (Blat 2bit data can also be in organism_data in directory 'searchDatabaseData')
- **genus** (*str*) – Genus
- **species** (*str*) – Species
- **public** (*bool*) – should the organism be public
- **non_default_translation_table** (*int*) – The translation table number for the organism (if different than that of the server's default)
- **metadata** (*str*) – JSON formatted arbitrary metadata

Return type dict

Returns a dictionary with information about the new organism

add_track(*organism_id*, *track_data*, *track_config*)

Adds a tarball containing track data to an existing organism.

The recommended structure for your track data tarball is as follows:

```
tracks/testing2/
tracks/testing2/Mijalis/
tracks/testing2/Mijalis/hist-2000-0.json
tracks/testing2/Mijalis/lf-1.json
tracks/testing2/Mijalis/lf-2.json
tracks/testing2/Mijalis/lf-3.json
tracks/testing2/Mijalis/names.txt
tracks/testing2/Mijalis/trackData.json
```

And an example of the track_config supplied at the same time:

```
{
    "key": "Some human-readable name",
    "label": "my-cool-track",
    "storeClass": "JBrowse/Store/SeqFeature/NCList",
    "type": "FeatureTrack",
    "urlTemplate": "tracks/testing2/{refseq}/trackData.json"
}
```

This is only the recommended structure, other directory structures / parameter combinations may work but were not tested by the python-apollo author who wrote this documentation.

Parameters

- **organism_id** (*str*) – Organism ID Number
- **track_data** (*file*) – .tar.gz or .zip archive containing the <track> directory.
- **track_config** (*dict*) – Track configuration

Return type dict

Returns a dictionary with information about all tracks on the organism

delete_organism(*organism_id*)

Remove an organism completely.

Parameters **organism_id** (*str*) – Organism ID Number

Return type dict

Returns a dictionary with information about the deleted organism

delete_track(*organism_id*, *track_label*)

Remove a track from an organism

Parameters

- **organism_id** (*str*) – Organism ID Number
- **track_label** (*str*) – Track label

Return type dict

Returns a dictionary with information about the deleted track

update_organism(*organism_id*, *organism_data*, *blatdb=None*, *common_name=None*, *genus=None*, *species=None*, *public=None*, *metadata=None*, *no_reload_sequences=False*)

Update an organism using the remote organism API.

The recommended structure for the genome data tarball is as follows:

```
names/
names/root.json
searchDatabaseData/blat_db.2bit
seq/
seq/fba/
seq/fba/da8/
seq/fba/da8/f3/
seq/fba/da8/f3/Mijalis-0.txt
seq/fba/da8/f3/Mijalis-1.txt
seq/fba/da8/f3/Mijalis-2.txt
seq/fba/da8/f3/Mijalis-3.txt
seq/fba/da8/f3/Mijalis-4.txt
seq/refSeqs.json
```

(continues on next page)

(continued from previous page)

```
tracks/
trackList.json
tracks.conf
```

The genome name / hashed directories below the seq folder will obviously be specific to your organism.

Parameters

- **organism_id** (*str*) – Organism ID Number
- **organism_data** (*file*) – .tar.gz or .zip archive containing the data directory.
- **blatdb** (*file*) – Path to 2bit index of the genome for Blat (Blat 2bit data can also be in organism_data in directory 'searchDatabaseData')
- **common_name** (*str*) – Organism common name
- **genus** (*str*) – Genus
- **species** (*str*) – Species
- **public** (*bool*) – User's email
- **metadata** (*str*) – JSON formatted arbitrary metadata
- **no_reload_sequences** (*bool*) – Set this if you don't want Apollo to reload genome sequences (no change in genome sequence)

Return type dict

Returns a dictionary with information about the updated organism

update_track(*organism_id*, *track_config*)

Update the configuration of a track that has already been added to the organism. Will not update data for the track.

And an example of the track_config supplied:

```
{
    "key": "Some human-readable name",
    "label": "my-cool-track",
    "storeClass": "JBrowse/Store/SeqFeature/NCList",
    "type": "FeatureTrack",
    "urlTemplate": "tracks/testing2/{refseq}/trackData.json"
}
```

Parameters

- **organism_id** (*str*) – Organism ID Number
- **track_config** (*dict*) – Track configuration

Return type dict

Returns a dictionary with information about all tracks on the organism

3.1.10 apollo.status package

Module contents

Contains possible interactions with the Apollo Status Module

class apollo.status.**StatusClient**(*webapolloinstance*, ***requestArgs*)

Bases: *apollo.client.Client*

CLIENT_BASE = '/availableStatus/'

add_status(*status*)

Add a status value

Parameters **status** (*str*) – New status

Return type dict

Returns A dictionnary containing status description

delete_status(*id_number*)

Delete a status

Parameters **id_number** (*int*) – status ID number

Return type dict

Returns an empty dictionary

get_statuses()

Get all statuses available in this Apollo instance

Return type list of dicts

Returns list of status info dictionaries

show_status(*status*)

Get a specific status

Parameters **status** (*str*) – Status to show

Return type dict

Returns A dictionnary containing status description

update_status(*id_number*, *new_value*)

Update a status name

Parameters

- **id_number** (*int*) – status ID number
- **new_value** (*str*) – The new status name

Return type dict

Returns an empty dictionary

3.1.11 apollo.users package

Module contents

Contains possible interactions with the Apollo Users Module

class apollo.users.**UsersClient**(*webapolloinstance*, ***requestArgs*)

Bases: *apollo.client.Client*

CLIENT_BASE = '/user/'

activate_user(*user*)

Activate a user

Parameters **user** (*str*) – User's email

Return type dict

Returns an empty dictionary

add_to_group(*group*, *user*)
Add a user to a group

Parameters

- **user** (*str*) – User's email
- **group** (*str*) – Group name

Return type dict

Returns an empty dictionary

create_user(*email*, *first_name*, *last_name*, *password*, *role='user'*, *metadata={}*)
Create a new user

Parameters

- **email** (*str*) – User's email
- **first_name** (*str*) – User's first name
- **last_name** (*str*) – User's last name
- **password** (*str*) – User's password
- **role** (*str*) – User's default role, one of "admin" or "user"
- **metadata** (*dict*) – User metadata

Return type dict

Returns an empty dictionary

delete_user(*user*)
Delete a user

Parameters **user** (*str*) – User's email

Return type dict

Returns an empty dictionary

get_organism_permissions(*user*)
Display a user's organism permissions

Parameters **user** (*str*) – User's email

Return type dict

Returns a dictionary containing user's organism permissions

get_user_creator(*user*)
Get the creator of a user

Parameters **user** (*str*) – User Email

Return type dict

Returns a dictionary containing user information

get_users(*omit_empty_organisms=False*)
Get all users known to this Apollo instance

Parameters **omit_empty_organisms** (*bool*) – Will omit users having no access to any organism

Return type list of dicts

Returns list of user info dictionaries

inactivate_user(*user*)
Activate a user

Parameters **user** (*str*) – User's email

Return type dict

Returns an empty dictionary

remove_from_group(*group*, *user*)
Remove a user from a group

Parameters

- **user** (*str*) – User's email
- **group** (*str*) – Group name

Return type dict

Returns an empty dictionary

show_user(*user*)
Get a specific user

Parameters **user** (*str*) – User Email

Return type dict

Returns a dictionary containing user information

update_organism_permissions(*user*, *organism*, *administrate=False*, *write=False*, *export=False*, *read=False*)
Update the permissions of a user on a specified organism

Parameters

- **user** (*str*) – User's email
- **organism** (*str*) – organism common name
- **administrate** (*bool*) – Grants administrative privileges
- **write** (*bool*) – Grants write privileges
- **read** (*bool*) – Grants read privileges
- **export** (*bool*) – Grants export privileges

Return type dict

Returns a dictionary containing user's organism permissions

update_user(*email*, *first_name*, *last_name*, *password=None*, *metadata={}*, *new_email=None*)
Update an existing user

Parameters

- **email** (*str*) – User's email
- **first_name** (*str*) – User's first name
- **last_name** (*str*) – User's last name

- **password** (*str*) – User's password (omit to keep untouched)
- **metadata** (*dict*) – User metadata
- **new_email** (*str*) – User's new email (if you want to change it)

Return type dict

Returns a dictionary containing user information

3.2 Submodules

3.3 apollo.client module

Base apollo client

class apollo.client.**Client**(*webapolloinstance*, ***requestArgs*)

Bases: object

Base client class implementing methods to make requests to the server

CLIENT_BASE = '/'

get(*client_method*, *get_params*, *is_json=True*)

Make a GET request

post(*client_method*, *data*, *post_params=None*, *is_json=True*, *files=None*, *autoconvert_to_json=True*)

Make a POST request

3.4 apollo.decorators module

3.5 apollo.exceptions module

exception apollo.exceptions.**APIErrorResponseException**

Bases: Exception

exception apollo.exceptions.**UnknownUserException**

Bases: Exception

3.6 apollo.util module

apollo.util.**AssertAdmin**(*user*)

apollo.util.**AssertUser**(*user_list*)

apollo.util.**CnOrGuess**(*parser*)

apollo.util.**GuessCn**(*args*, *wa*)

apollo.util.**GuessOrg**(*args*, *wa*)

apollo.util.**OrgOrGuess**(*parser*)

apollo.util.**WAAuth**(*parser*)

apollo.util.**add_property_to_feature**(*feature*, *property_key*, *property_value*)

Parameters

- **feature** –
- **property_key** (*str*) –
- **property_value** (*str*) –

Returns

apollo.util.**features_to_apollo_schema**(*features*, *use_name=False*, *disable_cds_recalculation=False*)

Parameters

- **disable_cds_recalculation** –
- **use_name** –
- **features** –

Returns

apollo.util.**features_to_feature_schema**(*features*, *use_name=False*, *disable_cds_recalculation=False*)

Parameters

- **disable_cds_recalculation** –
- **use_name** –
- **features** –

Returns

apollo.util.**retry**(*closure*, *sleep=1*, *limit=5*)

Apollo has the bad habit of returning 500 errors if you call APIs too quickly, largely because of the unholy things that happen in grails.

To deal with the fact that we cannot send an addComments call too quickly after a createFeature call, we have this function that will keep calling a closure until it works.

apollo.util.**yieldApolloData**(*feature*, *use_name=False*, *disable_cds_recalculation=False*)

3.7 Module contents

class apollo.**ApolloInstance**(*url*, *username*, *password*)

Bases: object

apollo.**accessible_organisms**(*user*, *orgs*, *permission=None*)

Get the list of organisms accessible to a user, filtered by *orgs*

class apollo.**fakeTrans**(*username*)

Bases: object

get_user()

apollo.**galaxy_list_groups**(*trans*, **args*, ***kwargs*)

apollo.**galaxy_list_orgs**(*trans*, **args*, ***kwargs*)

class apollo.**obj**

Bases: object

apollo.**require_user**(*wa*, *email*)
Require that the user has an account

apollo.**set_logging_level**(*level*)

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index

A

C

D

F

G

I

L

M

O

P

R

S

T

U

W

Y