BioMAJ Documentation

Release 3.0

Olivier Sallou

Aug 29, 2022

Contents

1 Getting Started **3**
- 1.1 Docker . . . 3
- 1.2 Simple Configuration . . . 3
- 1.3 Advanced Configuration . . . 5

2 Advanced Topics **7**
- 2.1 LDAP . . . 7
- 2.2 ElasticSearch . . . 7

3 bank **11**
- 3.1 bank API reference . . . 11

4 options **13**
- 4.1 Options API reference . . . 13

5 Session **15**
- 5.1 Session API reference . . . 15

6 workflow **17**
- 6.1 Workflows API reference . . . 17

7 notify **19**
- 7.1 Notify API reference . . . 19

8 metaprocess **21**
- 8.1 MetaProcess API reference . . . 21

9 processfactory **23**
- 9.1 ProcessFactory API reference . . . 23

10 Indices and tables **25**

Python Module Index **27**

Index **29**

Getting Started Documentation:

CHAPTER 1

Getting Started

For a very basic setup, you can configure a `docker-compose.yml` file to use with docker, which is especially helpful when you are testing out BioMAJ.

1.1 Docker

```
1  version: '2'
2  services:
3      biomaj:
4          image: osallou/biomaj-docker
5          links:
6              - mongodb:biomaj-mongodb
7          volumes:
8              - ./data:/var/lib/biomaj
9
10     mongodb:
11         image: mongo
```

This configuration file defines a simple MongoDB instance which is used for backend storage by BioMAJ, as well as the BioMAJ instance itself. Line 8 denotes that a folder named `data` in the current directory will be mounted into the volume as storage. Any files downloaded by BioMAJ will appear in this directory.

Running the `--help` command can be done easily:

```
$ docker-compose run --rm biomaj --help
```

1.2 Simple Configuration

Once you've reached this point, you're ready to start configuring BioMAJ to download datasets for you. Configuration files should go instead a folder `conf` inside the `data` folder in your current directory. As an example, we will use this simple ALU configuration file:

```
1  [GENERAL]
2  # Database name/description
3  db.fullname="alu.n : alu repeat element. alu.a : translation of alu.n repeats"
4  # The short name for the database
5  db.name=alu
6  # Database type. Some common values include genome, nucleic, nucleic_protein, protein,
   ↪ other
7  db.type=nucleic_protein
8  # Base directory to download to download temp files to
9  offline.dir.name=offline/ncbi/blast/alu_tmp
10 # Base directory to download to
11 dir.version=ncbi/blast/alu
12 # Update frequency
13 frequency.update=0
14 # Number of threads used during downloading
15 files.num.threads=1
16
17 # Protocol, common values include ftp, http
18 protocol=ftp
19 # The FQDN of the server you with to connect to
20 server=ftp.ncbi.nih.gov
21 # And the directory on that server
22 remote.dir=/blast/db/FASTA/
23 # The files to find in that page of the remote server.
24 remote.files=^alu.*\.gz$
25
26 # BioMAJ can automatically extract the version number from a release
27 # document. This will be covered in another section.
28 release.file=
29 release.regexp=
30 release.file.compressed=
31
32 #Uncomment if you don't want to extract the data files.
33 #no.extract=true
34
35 # ?
36 local.files=^alu\.(a|n).*
37
38 ## Post Process  ##  The files should be located in the projectfiles/process directory
39 db.post.process=
40
41 ### Deployment ###
42 keep.old.version=1
```

The file can be broken down into a couple of sections:

- Metadata (lines 1-15)
- Remote Source (17-24)
- Release Information (26-30)
- Other

The metadata consists of things like where data should be stored, and how to name it. The remote source describes where data is to be fetched from, release information we will see in another example, and then there are a few extra, miscellaneous options shown in this example config.

If you have copied the `alu.properties` file into `./data/conf/alu.properties`, you are ready to download this database:

```
$ docker-compose run --rm biomaj --bank alu --update
2016-08-24 21:43:15,276 INFO  [root][MainThread] Log file: /var/lib/biomaj/log/alu/
↪1472074995.28/alu.log
Log file: /var/lib/biomaj/log/alu/1472074995.28/alu.log
...
```

This command should complete successfully, and you will have some more files in `./data/`:

```
$ find data
data/conf/alu.properties
data/data/ncbi/blast/alu/alu-2003-11-26/flat/alu.a
data/data/ncbi/blast/alu/alu-2003-11-26/flat/alu.n
data/cache/files_1472074995.29
data/log/alu/1472074995.28/alu.log
```

The `data/data` directories contain your downloaded files. Additionally a cache file exists and a job run log is contains data about what occurred during the download and processing. Note that the files that appear are `alu.a` and `alu.n`, instead of `alu.a.gz` and `alu.n.gz`. By having the option `no.extract=true` commented out on line 33, BioMAJ automatically extracted the data for us.

The `--status` command will allow you to see the status of various databases you have downloaded.

```
$ docker-compose run --rm biomaj --bank alu --status
+--------+-----------------+----------------------+---------------------+
| Name   | Type(s)         | Last update status   | Published release   |
|--------+-----------------+----------------------+---------------------|
| alu    | nucleic_protein | 2016-08-24 21:58:14  | 2003-11-26          |
+--------+-----------------+----------------------+---------------------+
+---------------------+------------------+------------+------------------------------
↪---------------------+----------+
| Session             | Remote release   | Release    | Directory                    ␣
↪                     | Freeze   |
|---------------------+------------------+------------+------------------------------
↪---------------------+----------|
| 2016-08-24 21:58:14 | 2003-11-26       | 2003-11-26 | /var/lib/biomaj/data/ncbi/
↪blast/alu/alu-2003-11-26 | no       |
+---------------------+------------------+------------+------------------------------
↪---------------------+----------+
```

1.3 Advanced Configuration

Once you have this sort of simple configuration working, you may wish to explore more advanced configurations. There is a public repository of BioMAJ configurations which will be interesting to the advanced user wishing to learn more about what can be done with BioMAJ.

CHAPTER 2

Advanced Topics

2.1 LDAP

The BioMAJ watcher, provides an optional web interface to manage banks. Users can create "private" banks and manage them via the web.

2.2 ElasticSearch

In order to use the `--search` flag, you may wish to connect an ElasticSearch cluster.

You will need to edit your `global.properties` to indicate where the ES servers are:

```
use_elastic=0
#Comma separated list of elasticsearch nodes  host1,host2:port2
elastic_nodes=localhost
elastic_index=biomaj
# Calculate data.dir size stats
data.stats=1
```

An example `docker-compose.yml` would use this:

```
version: '2'
services:
    biomaj:
        image: osallou/biomaj-docker
        links:
            - mongodb:biomaj-mongodb
            - elasticsearch
        volumes:
            - ./data:/var/lib/biomaj
            - ./global.advanced.properties:/etc/biomaj/global.properties
```

(continues on next page)

(continued from previous page)

```
    mongodb:
        image: mongo

    elasticsearch:
        image: elasticsearch:1.7
```

And a modified `global.properties` referenced in that file would enable elasticsearch:

```
[GENERAL]
root.dir=/var/lib/biomaj
conf.dir=%(root.dir)s/conf
log.dir=%(root.dir)s/log
process.dir=%(root.dir)s/process
cache.dir=%(root.dir)s/cache
lock.dir=%(root.dir)s/lock
#The root directory where all databases are stored.
#If your data is not stored under one directory hirearchy
#you can override this value in the database properties file.
data.dir=%(root.dir)s/data

db.url=mongodb://biomaj-mongodb:27017
db.name=biomaj

use_ldap=0
ldap.host=localhost
ldap.port=389
ldap.dn=nodomain

use_elastic=1
#Comma separated list of elasticsearch nodes  host1,host2:port2
elastic_nodes=elasticsearch
elastic_index=biomaj
# Calculate data.dir size stats
data.stats=1

celery.queue=biomaj
celery.broker=mongodb://biomaj-mongodb:27017/biomaj_celery


auto_publish=1

########################
# Global properties file


#To override these settings for a specific database go to its
#properties file and uncomment or add the specific line you want
#to override.

#----------------
# Mail Configuration
#---------------
#Uncomment thes lines if you want receive mail when the workflow is finished

mail.smtp.host=
#mail.stmp.port=25
mail.admin=
```

(continues on next page)

(continued from previous page)

```
mail.from=biomaj@localhost
mail.user=
mail.password=
mail.tls=
# tail last X bytes of log in mail body , 0 = no tail
# mail.body.tail=2000000
# attach log file if size < X bytes, 0 for no attach
#mail.body.attach=4000000
# path to jinja template for subject, leave empty for defaults
#mail.template.subject=
# path to jinja template for body, leave empty for default
#mail.template.body=

#---------------------
#Proxy authentification
#---------------------
#proxyHost=
#proxyPort=
#proxyUser=
#proxyPassword=

#---------------------
# PROTOCOL
#--------------------
#possible values : ftp, http, rsync, local
port=21
username=anonymous
password=anonymous@nowhere.com



#access user for production directories
production.directory.chmod=775

#Number of thread during the download
bank.num.threads=4

#Number of threads to use for downloading and processing
files.num.threads=4

#to keep more than one release increase this value
keep.old.version=0

#Link copy property
do.link.copy=true


#The historic log file is generated in log/
#define level information for output : DEBUG,INFO,WARN,ERR
historic.logfile.level=INFO

http.parse.dir.line=<a[\\s]+href=\"([\\S]+)/\".*alt=\"\\[DIR\\]\">.*([\\d]{2}-[\\w\\d]
↪{2,5}-[\\d]{4}\\s[\\d]{2}:[\\d]{2})
http.parse.file.line=<a[\\s]+href=\"([\\S]+)\".*([\\d]{2}-[\\w\\d]{2,5}-[\\d]{4}
↪\\s[\\d]{2}:[\\d]{2})[\\s]+([\\d\\.]+[MKG]{0,1})

http.group.dir.name=1
```

(continues on next page)

(continued from previous page)

```
http.group.dir.date=2
http.group.file.name=1
http.group.file.date=2
http.group.file.size=3

#Needed if data sources are contains in an archive
log.files=true

local.files.excluded=\\.panfs.*

#~40mn
ftp.timeout=2000000
ftp.automatic.reconnect=5
ftp.active.mode=false

# Bank default access
visibility.default=public

#proxy=http://localhost:3128

[loggers]
keys = root, biomaj

[handlers]
keys = console

[formatters]
keys = generic

[logger_root]
level = INFO
handlers = console

[logger_biomaj]
level = INFO
handlers = console
qualname = biomaj
propagate=0

[handler_console]
class = StreamHandler
args = (sys.stderr,)
level = DEBUG
formatter = generic

[formatter_generic]
format = %(asctime)s %(levelname)-5.5s [%(name)s][%(threadName)s] %(message)s
```

API Documentation:

CHAPTER 3

bank

3.1 bank API reference

CHAPTER 4

options

4.1 Options API reference

class biomaj.options.**Options**(*options=None*)
Available options

__init__(*options=None*)
x.__init__(...) initializes x; see help(type(x)) for signature

__weakref__
list of weak references to the object (if defined)

get_option(*option*)
Gets an option if present, else return None

CHAPTER 5

Session

5.1 Session API reference

CHAPTER 6

workflow

6.1 Workflows API reference

CHAPTER 7

notify

7.1 Notify API reference

CHAPTER 8

metaprocess

8.1 MetaProcess API reference

class biomaj.process.metaprocess.**MetaProcess**(*bank*, *metas*, *meta_status=None*, *meta_data=None*, *simulate=False*)

Meta process in biomaj process workflow. Meta processes are executed in parallel.

Each meta process defined a list of Process to execute sequentially

__init__(*bank*, *metas*, *meta_status=None*, *meta_data=None*, *simulate=False*)

Creates a meta process thread

Parameters

- **bank** (biomak.bank) – Bank
- **meta** (*list of str*) – list of meta processes to execute in thread
- **meta_status** (*bool*) – initial status of the meta processes
- **simulate** (*bool*) – does not execute process

_get_metata_from_outputfile(*proc*)

Extract metadata given by process on stdout. Store metadata in self.metadata

Parameters **proc** – process

run()

Method representing the thread's activity.

You may override this method in a subclass. The standard run() method invokes the callable object passed to the object's constructor as the target argument, if any, with sequential and keyword arguments taken from the args and kwargs arguments, respectively.

set_progress(*name*, *status=None*)

Update progress on execution

Parameters

- **name** (*str*) – name of process
- **status** (*bool or None*) – status of process

CHAPTER 9

processfactory

9.1 ProcessFactory API reference

class biomaj.process.processfactory.**PostProcessFactory**(*bank*, *blocks=None*, *redis_client=None*, *redis_prefix=None*)

Manage postprocesses

self.blocks: dict of meta processes status Each meta process status is a dict of process status

__init__(*bank*, *blocks=None*, *redis_client=None*, *redis_prefix=None*)

Creates a postprocess factory

Parameters

- **bank** (biomaj.bank.Bank) – Bank
- **blocks** (*dict*) – initial status of block processes

run(*simulate=False*)

Run processes

Parameters **simulate** (*bool*) – does not execute process

Returns status of execution - bool

class biomaj.process.processfactory.**PreProcessFactory**(*bank*, *metas=None*, *redis_client=None*, *redis_prefix=None*)

Manage preprocesses

__init__(*bank*, *metas=None*, *redis_client=None*, *redis_prefix=None*)

Creates a preprocess factory

Parameters

- **bank** (biomaj.bank.Bank) – Bank

- **metas** (*dict*) – initial status of meta processes

run(*simulate=False*)
Run processes
Parameters **simulate** (*bool*) – does not execute process
Returns status of execution - bool

class biomaj.process.processfactory.**ProcessFactory**(*bank*, *redis_client=None*, *redis_prefix=None*)
Manage process execution

__init__(*bank*, *redis_client=None*, *redis_prefix=None*)
x.__init__(...) initializes x; see help(type(x)) for signature

__weakref__
list of weak references to the object (if defined)

fill_tasks_in_threads(*metas*)
Dispatch meta processes in available threads

run(*simulate=False*)
Run processes
Parameters **simulate** (*bool*) – does not execute process
Returns status of execution - bool

run_threads(*simulate=False*)
Start meta threads
Parameters **simulate** (*bool*) – do not execute processes
Returns tuple global execution status and status per meta process

class biomaj.process.processfactory.**RemoveProcessFactory**(*bank*, *metas=None*, *redis_client=None*, *redis_prefix=None*)
Manage remove processes

__init__(*bank*, *metas=None*, *redis_client=None*, *redis_prefix=None*)
Creates a remove process factory
Parameters
- **bank** (biomaj.bank.Bank) – Bank
- **metas** (*dict*) – initial status of meta processes

run(*simulate=False*)
Run processes
Parameters **simulate** (*bool*) – does not execute process
Returns status of execution - bool

CHAPTER 10

Indices and tables

- genindex
- modindex
- search

Python Module Index

b

biomaj.options, 13
biomaj.process.metaprocess, 21
biomaj.process.processfactory, 23

Index

Symbols

__init__() (*biomaj.options.Options method*), 13
__init__() (*biomaj.process.metaprocess.MetaProcess method*), 21
__init__() (*biomaj.process.processfactory.PostProcessFactory method*), 23
__init__() (*biomaj.process.processfactory.PreProcessFactory method*), 23
__init__() (*biomaj.process.processfactory.ProcessFactory method*), 24
__init__() (*biomaj.process.processfactory.RemoveProcessFactory method*), 24
__weakref__ (*biomaj.options.Options attribute*), 13
__weakref__ (*biomaj.process.processfactory.ProcessFactory attribute*), 24
_get_metata_from_outputfile() (*biomaj.process.metaprocess.MetaProcess method*), 21

B

biomaj.options (*module*), 13
biomaj.process.metaprocess (*module*), 21
biomaj.process.processfactory (*module*), 23

F

fill_tasks_in_threads() (*biomaj.process.processfactory.ProcessFactory method*), 24

G

get_option() (*biomaj.options.Options method*), 13

M

MetaProcess (*class in biomaj.process.metaprocess*), 21

O

Options (*class in biomaj.options*), 13

P

PostProcessFactory (*class in biomaj.process.processfactory*), 23
PreProcessFactory (*class in biomaj.process.processfactory*), 23
ProcessFactory (*class in biomaj.process.processfactory*), 24

R

RemoveProcessFactory (*class in biomaj.process.processfactory*), 24
run() (*biomaj.process.metaprocess.MetaProcess method*), 21
run() (*biomaj.process.processfactory.PostProcessFactory method*), 23
run() (*biomaj.process.processfactory.PreProcessFactory method*), 24
run() (*biomaj.process.processfactory.ProcessFactory method*), 24
run() (*biomaj.process.processfactory.RemoveProcessFactory method*), 24
run_threads() (*biomaj.process.processfactory.ProcessFactory method*), 24

S

set_progress() (*biomaj.process.metaprocess.MetaProcess method*), 21