

August 14, 2016



<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Thoughts on workflow organization</b>	<b>5</b>
<b>3</b>	<b>Table of contents:</b>	<b>7</b>
<b>4</b>	<b>Indices and tables</b>	<b>143</b>
	<b>Python Module Index</b>	<b>145</b>



*Command line workflows in bioinformatics*

Moa aims to assist a bioinformatician to organize, document, share, inspect, execute and repeat workflows in a command line environment - without losing any of the flexibility of the command line (see [Goals](#)).

Download and installation instructions can be found in: [Installation](#).

**\*NOTE: The software (and manual) are under development. Things might still change.\***



---

# Introduction

---

These days, generating massive amounts of data is an everyday element of biological research; and almost all projects have a computational biology, or bioinformatics, components. Such embedded work commonly consists of chaining a number of 3<sup>rd</sup> party tools together, often with some data manipulation in between the steps. It is important to have such projects properly organized, particularly when a projects grows bigger.

There are many different ways to organize a bioinformatics project. Many bioinformaticians use the command line, scripts or [Makefiles](#) to organize and automate their work. This approach has obvious advantages, most importantly flexibility. With almost any approach, meticulous care needs to be taken to keep a project well organized and documented. If this is not done, it is easy to lose track, certainly when others have to try to make sense of your project.

Moa hopes to make meticulous organization of a command line project much less of a burden - leaving you to focus on the fun parts.





---

### Thoughts on workflow organization

---

Most (bioinformatics?) projects start small, and grow over time. From that perspective it is advisable to give the organization of your project some thought on forehand.

When using Moa a workflow resides in a directory tree, with each directory containing the separate analysis steps. A Moa job is linked to a directory, and one directory can contain only Moa job. In- and output data of each analysis typically resides in the same directory structure. Having bot structure and data as regular files on your file system makes a workflow extremely accessible. It is however important that the directory structure represent the workflow in a logical manner.

There are likely multiple ways of achieving a healthy organization of a bioinformatics (Moa) project, we proposes the following organization:

- On the highest levels organize your project according to fundamental divisions in the project or data source. For example, if you work with data from multiple organisms, that might be a good top level division.
- On lower levels start organizing your annotation pipeline. Since most



---

## Table of contents:

---

### 3.1 Goals

The objective of Moa is to assist a bioinformatics project by keeping it;

- *Organized:*

Moa facilitates project organization in many (smaller and more major) ways by providing a uniform way to capture commands as Moa jobs. Each Moa job is linked to a specific directory, which contains all configuration, template, data, and intermediate data available as files within the directory structure.

- *Documented:*

Moa provides the possibility to add a title, description and changelogs to each job.

- *Reproducible*

By having all templates and configuration copied into a workflow - the workflow never changes (unless the user wants it to), even if templates in the repository change. Moreover, all templates are easy to find and inspect so it is always clear what happened.

- *Reusable & Shareable:*

Moa provides reusable templates. New templates are easy to create, adapt and share. Workflows can be archived and reused with different data.

- *Flexible:*

Moa provides a good number of hooks to insert custom code into a workflow, making that code part of the workflow. This ensures maximum flexibility.

### 3.2 Installation

#### 3.2.1 Prerequisites

Moa is developed and tested on [Ubuntu](#) and [RHEL](#) and is expected to operate without much problems on most modern Linux distributions. Moa requires [Python](#) (2.6 or 2.7). Moa will not work with earlier versions or with Python3.

Recommended prerequisites are:

- *python-dev*: The Python development libraries. A number of prerequisites installed by pip or easy\_install will try to compile C libraries, and need this. Although all have a backup, Python only, version - the performance of the C, performance will suffer. On a debian based distribution, call:

```
sudo apt-get install python-dev
```

While on RHEL flavoured distribution users might run:

```
sudo yum install -y python-devel
```

- ***python-yaml***: This will install a faster YAML parser, as opposed to the python only YAML parser you would probably get when installing through pip or easy-install. On a debian based distro:

```
sudo apt-get install python-yaml
```

While on RHEL flavoured distribution users will find this in the [EPEL](#) repository and might want to run:

```
sudo yum install -y pyyaml
```

## Git integration

One feature of Moa is the ability to integrate with [Git](#) to keep track of your workflow. If you want to use this, you (obviously) need Git installed. For most applications the package manager version is fine. However, Moa is able to pull templates from git repositories. If you want to use that feature, you must install *git subtree*. This application comes bundled with recent version of Git (certainly with 1.8) but still needs to be installed separately. Otherwise, it can be downloaded from the “[apenwarr](#)” repository.

### 3.2.2 Blue Ringed Octopus

Blue Ringed Octopus is a (randomly named) repository with a number of helper scripts, used by a number of templates. You might want to install this - just check out the repository and either add it to your PATH, or copy the scripts to a location in the PATH. The repository can be found here:

```
https://github.com/mfiers/Blue-Ringed-Octopus
```

### 3.2.3 Installation of Moa

It is most convenient to install Moa from the [Python package index](#):

```
pip install Moa
```

(You might need root rights to do this, also - pip is similar to easy\_install, so if you want you can run *easy\_install Moa*)

You will definitely need [pip](#) installed to run the pip command above which is a replacement for easy\_install.

Note that it is possible, and even recommended, to install Moa within a [virtual environment](#).

Moa should now work, try *moa -help*.

## Bash integration

Moa comes with a number of functions to improve integration with Bash. To turn these on, execute the following command (or add this to your `~/.bashrc`):

```
source $(moainit)
```

This does a number of things:

- adds an alias *msh* for *moa set process*
- adds tab completion
- records a bash history for each separate moa job, equivalent to your bash history, but stored with your job.

**Note that this is a possible privacy concern.** Commands that are not related to your workflow will be recorded (and possibly shared) as well. If you want to remove your history, delete `.moa/local_bash_history`. For a complete workflow run (in the root of that workflow):

```
find . -name local_bash_history | xargs rm
```

The `local_bash_history` is, however, not tracked by the Git module (unless specified explicitly)

### 3.2.4 Manual installation (from Github)

When installing manually, you'll need the following prerequisites:

- `pyyaml`
- `argparse` (only for Python2.6)
- `Jinja2`
- `ruffus`
- `unittest2`
- `lockfile`
- `GitPython`
- `Yaco`
- `fist`

Once these are installed, you can get Moa from, [Github](#). Run the following command (in an appropriate location):

```
git clone git://github.com/mfiers/Moa.git
```

To install Moa, run:

```
cd Moa
python setup.py install
```

If this is for a global installation, you'll need to be root, or use `sudo`.

Moa should now work, try `moa -help`.

### 3.2.5 Troubleshooting

A potential problem could be that your python version is NOT *python2.6* or *python2.7* there are a few options that you can pursue:

- Make sure python2.6 or 2.7 is installed.
- define an alias in your `~/.bashrc`: `alias moa='python2.7 moa'`
- create a symlink to python2.7 in your `~/bin` directory and make sure that that is first in your path
  - but note that this will change the Python version for all you user scripts.

### 3.2.6 Bioinformatics tools

Each of the wrapped tools requires the tools to be present. Usually, Moa expects all tools to be present & executable on the system PATH. The standard Moa distribution comes with wrappers for Blast, BWA and Bowtie. Note that a number of tools also depends on [Biopython](#).

## 3.3 Tutorial 1

(note, to fully use the blast template - you will need the *blastReport* script from the [Blue Ringed Octopus](#) repository).

This quick start aims to help you understand how Moa can help to organize a command line bioinformatics project.

Each Moa workflow consists of separate Moa jobs. An important feature of Moa is that each Moa job resides in a directory, and each directory can hold only *one* Moa job. A workflow is organised as a directory tree, where the structure of the directory tree reflects the structure of the project. This will (hopefully) stimulate a user to break a workflow down into atomic parts, which will be beneficial to the organization and coherence of a workflow. So, starting a Moa project starts with creating a directory to hold the workflow:

```
$ mkdir test.project
$ cd test.project
$ mkdir 00.proteins

## copy some protein sequences in 00.proteins

$ mkdir 10.blast
$ cd 10.blast
```

The order of steps can be ordered by prefixing directory names with a number. Note that this not enforced by Moa. Once a directory is created, a Moa job can be created (see [moa new](#)):

```
$ moa new blast -t "demo run"
```

All interaction with Moa is done through a single command: *moa*. It is, at all times, possible to get help on the use of the *moa* command by invoking *moa -h* or *moa -help*. The command above creates a **BLAST** job titled “demo run” in the current directory. All Moa related files are stored in a (hidden) sub-directory named *.moa* (go and have a look!). A Moa job consists, amongst others, of a configuration file (*.moa/config*) and a number of template files (*.moa/template* and/or *.moa/template.d/\**). All template files are copied into the *.moa* directory. This ensures that a workflow remains the same over time, even

if the templates are updated. If you want to copy the latest version of a template to a Moa job, use *moa refresh*.

Moa also tries to assist in embedding documentation. In the above command line the *-t* parameter sets a mandatory project title (a job won't execute without a title). Moa also automatically records a changelog (in *.moa/doc/change*). You can add your own changelog messages by using the *-m* argument (before the command!) or by using *command\_moa\_change*. Additionally, you can keep a “blog” (*command\_moa\_blog*) for a higher level record on the development of the work, and a “readme” (*command\_moa\_readme*) to create a document for each job.

Back to the blast job - it is obviously not enough to tell Moa to do a BLAST analysis. Some extra information is necessary (see *moa set*):

```
$ moa set db=/data/blast/db/nr
```

A few points are important to note: do not use spaces around the = sign. If you want to define a parameter with spaces, use quotes (*key="value with spaces"*), and be very aware of bash expansion. A safer way to enter parameters is by running *moa set db* and Moa will query you for the value (note that in both cases you can use tab-completion).

If you want to check what the parameters are, you can use *moa show*, which will give you a list of parameters known to Moa:

```
$ moa show
db          1.M   /data/blast/db/nr
input       d.M   */*.fasta
jobid       s.o   blast
title       1.M   demo run
...
```

Note the variable *db* and *title*, which were set earlier. If you run *show -a*, more parameters will be revealed, amongst which is *program*. The flags between the variable key and value are explained in: *moa show*.

We will now set two more variables:

```
$ moa set program=blastp
$ moa set input=../00.proteins/*.fasta
```

The last statement defines the input files to blast. Once all is set you can actually run (see *moa run*) the BLAST analysis with:

```
$ moa run
```

Moa now performs the BLAST analysis on each of the input files. The output can be found in the *out* sub-directory. As an extra, the Moa *blast* template generates a *blast\_report* file with simple one line report for the best five hits of each query sequence.

To illustrate how easy it is to embed extra command lines into your workflow, we will check for the presence of any *dicer* genes in the query set by employing *grep*:

```
$ grep -i dicer blast_report
```

To embed this in the workflow, execute:

```
$ moa set postcommand
```

and, at the prompt enter:

```
postcommand:
> grep -i dicer blast_report > dicer.out
```

If you now rerun *moa*, the BLAST job will not be repeated, but the *postcommand* will be executed and a *dicer.out* file will be generated. (note, there is also a *precommand*)

If this is all clear, continue with [Tutorial 2](#).

## 3.4 Tutorial 2

Note, you will need to have *git* and *git subtree* installed for this tutorial.

## 3.5 Configuring Moa

Moa is configured using the command line tool *moa*. For example, you are creating a simple job somewhere:

```
$ moa simple -t 'test job' -- echo "Hello"
```

and you would like to change the title, you can do this with *moa set*:

```
$ moa set title='I mean: Hello!!'
```

When setting parameters on the command line you need to consider the fact that bash might try to expand or interpret the command line. For example, if you would like to set the process parameter to *echo “it’s complicated”*, you would need the following command line:

```
$ moa set process='echo "it'\'' complicated"'
```

only to be able to use a single quote. Similar care needs to be taken with, for example, the *\$* character, as that will be expanded by bash, unless placed between single quotes or properly escaped. An alternative way of setting variables would be by running:

```
$ moa set process
```

which will prompt you for the value of *process*, without trying to expand any variables.

It is at all times possible to check what the current configuration is by running:

```
$ moa show
```

which will give you (possibly with more color):

```
postcommand  o (undefined)
precommand   o (undefined)
process       L echo "it's complicated"
project       o (undefined)
title         L I mean: Hello!!
```

The first column has the parameter name, followed by a single letter, and the value of the parameter, or (*undefined*) if no value is specified. The letters in the second column signify the state of the parameter:

- **o**: Undefined, but **O**ptional.
- **L**: Locally defined



- **R:** Recursively defined
- **E:** Error - undefined and not optional.

## 3.6 Execution

What is executed upon a Moa run can either be defined by a plugin, or by a template. Most Moa commands (such as *moa show* are plugin defined). Only *moa prepare*, *moa run* and *moa finish* call code defined in a template. A number of steps are:

### 3.6.1 Main invocation

The complete Moa Invocation is embedded in a try / except.

On an error, Moa tries to execute a *post\_error* hook and then attempts to fail quietly. If you are interested in the actual error, run moa with the '-v' flag

Upon a keyboard interrupt, Moa executes the *post\_interrupt* hook and exists with a return code of -2.

### 3.6.2 Background execution

The first thing Moa does is to check if '-bg' is defined on the command line? If so, fork, let the child thread continue and let parent thread exit.

Before continuation, the parent thread executes the *background\_exit* hook before exit. The child thread executes the *post\_background* hook before continuing.

### 3.6.3 Recursive execution

Moa **used** to have the '-r' flag for all operations, allowing recursive operation of Moa. This was rather confusing and has been removed. Some commands still define -r (such as 'moa cp'), but for the majority of commands, you will need to use bash (find, xargs, etc), or use the new, stand-alone, helper script 'moar'. Using 'moar' is very simple:

```
moar -- moa run
```

runs 'moa run' in this directory, and all (non hidden) sub-directories. If you would like to limit execution to a certain depth, for example only first level sub-directories, you can run:

```
moar -d 1 -- moa run
```

## 3.7 Filesets

Filesets are an important part of Moa - they are used to define input and output files for Moa jobs. In principle, a fileset is not much more than a collection of files. They are three different types:

### 3.7.1 Types

#### Type “set”

A “set” fileset is given a filesystem [glob](#), checks the filesystem and returns a list of files that conform to the glob pattern. Type “set” filesets are typically used to define input of a Moa job. A “set” fileset can (currently) contain only one \* wildcard. A correct example would be:

```
/data/sequences/*.fasta
```

This glob does exactly what you expect. Lets assume that there are three sequences in this directory, the set would contain three filenames:

```
/data/sequences/input_01.fasta  
/data/sequences/input_02.fasta  
/data/sequences/input_03.fasta
```

More complex patterns, and wildcards other than \* are not supported (yet). Each Moa job can have at most one “set” fileset.

#### Type “map”

A “map” fileset converts a “set” fileset (the source) to a related fileset, typically to calculate the output of Moa job. A “map” fileset must be linked to “set” fileset and uses a glob like pattern to convert the input “set” fileset to the resulting fileset. For example, if we take the example fileset defined above, and apply the following pattern:

```
./*.output
```

we would end up with the following “map” fileset:

```
./input_01.output  
./input_02.output  
./input_03.output
```

A potential pitfall is the following situation, where we have a “set” fileset defined as follows:

```
/data/sequences/input_*.fasta
```

This would result in exactly the same fileset as above. But if we now apply the same “map” pattern, the resulting output fileset would be:

```
./01.output  
./02.output  
./03.output
```

This is because the \* from the “set” glob maps the the \* in the “map” pattern, the rest is omitted. This can be useful, for example if you would be using this in a Blast job, you could specify the following “map” pattern:

```
./blast_*.out
```

which would result in the following output:

```
./blast_01.out  
./blast_02.out  
./blast_03.out
```

In the case of a “map” set it is allowed to use a second wildcard in the pattern, for example:

```
*/blast_*.out
```

in which case the first wildcard is replaced with the original path. In the above example this would result in:

```
/data/sequences//blast_01.out
/data/sequences//blast_02.out
/data/sequences//blast_03.out
```

(note . you might not want to do this)

## Type “single”

Is a very simple fileset, pointing to a single file. No wildcards are allowed.

### 3.7.2 Categories

Moa has to keep track (using Ruffus) of in- and output of a job - it does this by tracking filesets. The category defines in a file(set) is considered “input”, “output” or a “prerequisite”. In- & output speaks for itself, a prerequisite is also considered input (i.e. if it changes the job will be repeated), but is typically kept out of the one-on-one file mapping that takes place for in- and output files.

### 3.7.3 Defining filesets

If you are developing a template, there is whole section devoted to filesets. The following example is taken from the Moa BLAST template, and contains almost everything that you will come across:

```
filesets:
  db:
    category: prerequisite
    help: Blast database
    optional: false
    pattern: '*/*'
    type: single
  input:
    category: input
    help: Directory with the input files for BLAST, in Fasta format
    optional: false
    pattern: '*/*.fasta'
    type: set
  outgff:
    category: output
    help: GFF output files
    optional: true
    pattern: gff/*.gff
    source: input
    type: map
  output:
    help: XML blast output files
    category: output
    optional: true
    pattern: out/*.out
```

```
source: input
type: map
```

Most of this speaks for itself. A few things to note are:

- Both “outff” and “output” are category “output”, type “map”, filesets mapping to the same input, type “set”, fileset. This is common practice. If you have a look at the `map22` template, you can even see an example of category “input”, type “map” fileset.
- If a fileset has reasonable default patterns (values) (typically goes for output fileset), it is possible to make them optional.
- Please specify a good help text

## 3.8 Three core templates

Moa comes with a list of templates (see `templates`). The three most important, flexible templates of these that allow you to embed custom code (called *process*) in your project are:

*simple*:

Simply executes *process* as a bash one-liner

*map*:

Takes a set of in- and output files and executes the custom commands for each in- and output file (using the [Jinja2](#) template language).

*reduce*:

Takes a set of input files and a single output file and executes the custom commands with all input file, generating the output files.

Since *simple*, *map* and *reduce* have proven to be quite central to how Moa operates they come with their own shortcut commands (*moa simple*, *moa map* and *moa reduce*). These command query the user directly for the parameters instead of having to define this manually.

For example, a *simple* job:

```
$ mkdir simple_test && cd simple_test
$ moa simple -t 'Generate some files'
process:
> for x in `seq 1 5`; do touch test.$x; done
$ moa run
$ ls
test.1 test.2 test.3 test.4 test.5
```

Note that you can make your *process* as complicated as you like. Alternatively, you can write a script that you call from *process*.

A *map* job would work like this:

```
$ mkdir ../map_test && cd ../map_test
$ moa map -t 'Map some files'
process:
> echo {{ input }} ; echo {{ input }} > {{ output }}
input:
> ../simple_test/test.*
```

```

output:
> ./out.*
$ moa run
../simple_test/test.3
../simple_test/test.1
../simple_test/test.5
../simple_test/test.2
../simple_test/test.
Moa: Success executing "run" (<1 sec)
$ ls
out.1  out.2  out.3  out.4  out.5
$ cat out.1
../simple_test/test.1

```

Moa tracks which input file generates which outputfile. So, if you would like to repeat one of the jobs - you'll need to delete the output file & rerun *moa*:

```

$ rm out.3
$ moa run
../simple_test/test.3
Moa: Success executing "run" (<1 sec)

```

And a *reduce* example:

```

$ mkdir ../reduce_test && cd ../reduce_test
$ moa reduce -t 'Reduce some files'
process:
> echo {{ input|join(" ") }} >> {{ output }}
input:
> ../map_test/out.*
output:
> ./reduce_out
$ moa run
Moa: Success executing "run" (<1 sec)
$ ls
reduce_out
$ cat reduce_out
../map_test/out.1 ../map_test/out.3 ../map_test/out.4 ../map_test/out.5 ../map_test/out.

```

## 3.9 Synchronizing jobs

It is quite often usefull to repeat a jobs on a number of different input files. For simple operations, one liners, this ca be accomplished using *moa map*. More complex operations, or those requiring a template other than *map* can be replicated using job synchronization. Assume you have a set of fastq libraries, each in it's own directory:

```

./fq/set1/set1_1.fq
./fq/set1/set1_2.fq
./fq/set2/set2_1.fq
./fq/set2/set2_2.fq
./fq/set3/set3_1.fq
./fq/set3/set3_2.fq

```

And you want to run a bowtie alignment for each separately. The approach to take is to create a directory containing all alignments:

```
mkdir bowtie
cd bowtie
```

and, in that directory, create one job running bowtie, in a directory named **exactly** as the input directories:

```
mkdir set1
cd set1
moa new bowtie -t 'run bowtie for {{_}}'
```

Note the magic variable `{{_}}`. This variable is replaced by the name of the current directory. So when running `moa show`, the title would show up as “run bowtie for set1”. This magic variable can be used in all variables, and we’ll use it here to set this job up in such a way that it can be reused for the other datasets:

```
set moa fq_forward_input='../../fq/{{_}}/*_1.fq'
# .. configure the remaining variables
```

Now - we replicate this directory in the following manner. We’ll move one directory up, to the *bowtie* directory, and create a *sync* job:

```
cd ..
moa new sync -t 'run bowtie for all fq datasets'
moa set source=../fq/
```

The sync template keeps directories synchronized, based on the *source* directory. If you now run `moa run` in the *bowtie* directory, two more directories will be created: *set2* and *set3*, each containing a verbatim copy of the original bowtie job created.

If, at a certain moment you obtain more fastq datasets:

```
../fq/set4/set4_1.fq
../fq/set4/set4_2.fq
```

you can repeat `moa run` in the *./bowtie* sync directory, and a new directory will be created. Note that the *sync* template will not remove directories. Also if you want to update the configuration of the synchronized bowtie jobs, you only need to change the configuration in one directory, run `moa run` again in the *./bowtie* directory and the configuration is synchronized across all jobs.

## 3.10 Git integration

Note: \* Integration with Git is a relatively new feature - there might be dragons.

- Moa/Git will try to keep track of the *structure* of your workflow, **not** of the data you are processing.

Moa integrates with [Git](#) for a number of reasons:

- to automatically keep your workflow under version control. Having your work under version control means that at all times you can find out what your workflow looked like at a certain date.
- to allow you to share your workflow using Git. For example, you could publish your workflow to Github, and allow other researchers to clone and improve on your work (which you can then import into your own workflow again).
- to pull templates from remote git repositories. This allows anybody to create, maintain and distribute templates for you to use.

To make this work, you must make sure that *git* and *git subtree* are installed and that the *moaGit* plugin is enabled. Also make sure that your workflow is under Git version control. If you create 'Moa projects', Moa will try to create a new repository for you. Otherwise, you must run *git init* to create a new repository.

### 3.10.1 A workflow under Git control

If you've created a new Moa project & made sure it is under git control, Moa will try to automatically commit all changes. One important thing to notice here is that if you make manual changes to the workflow - you will need to commit them yourself. If you fail to do so, they will likely be automatically committed by the next Moa operation. In which case they are under version control, but the commit message will not make any sense.

### 3.10.2 Sharing a workflow using Git

Your workflow is a normal git repository. See the excellent documentation of Git & Github how to share git repositories.

### 3.10.3 Getting templates from a remote git repository

If you want to install a template from a remote git repository, Moa will merge the template repository with the workflow's repository using *git subtree*. This approach has a number of nice properties:

- The template is integrated in the local workflow and can be copied around and changed (as one normally would do within a git repository).
- The template code does not change unless requested (as with regular templates). Even when sharing or duplicating your repository - the template remains unchanged.
- If required, it is possible to update to the latest version of the remote template repository (using a regular *moa refresh*)
- It possible to upload the template changes upstream.

Note - the git submodule approach was another candidate for implementing this, but submodules are difficult to be copied once they are checked out. Additionally, the *git subtree* approach has as an advantage above the *git subtree merge strategy* that it is easier to upload changes upstream.

### Define a template provider

To set up moa to work with git template you need to define a template provider, for example:

```
$ moa set -s template.providers.gtp.class=gitmodule
$ moa set -s template.providers.gtp.base='https://github.com/mfiers/moa_template_%s'
$ moa set -s template.providers.gtp.enabled=true
```

This defines a provider (called gtp) that pulls templates from github (but any other git server can be used). Note that it is probably advisable to set *\*.enabled=true* as last - to prevent an incomplete import.

After doing this you can run:

```
$ mkdir 30.run_bowtie
$ cd 30.run_bowtie
$ moa new gtp:bowtie -t 'a sensible title'
```

which would expect and merge a repository in the following location:

[https://github.com/mfiers/moa\\_template\\_bowtie](https://github.com/mfiers/moa_template_bowtie)

Moa requires to have template name (bowtie in the example above) and the configured *base* resolve to a valid git repository url. This provides a user friendly syntax and the ability to use any git repository required.

## 3.11 Contribute

Any contribution is more than welcome!

### 3.11.1 Documentation

You can find the documentation source on Github [Moa repository](#) in the [Sphinx](#) subdirectory. The documentation is formatted in [Restructured Text](#) and generated using [Sphinx](#).

If you would like to work on the documentation, you can clone the repository and send pull requests for any change you make. However, there is an easier way. Make sure you have a github account first. Once you're logged into Github, go to the moa [sphinx](#) subdirectory, click on the \*.rst file you would like to edit and subsequently click the edit button. You now get a warning that Github is forking the repository - which is fine - there should be an edit window that allows you to edit the text. Once you're done editing, please write a little (commit) message describing what you changed, and if necessary, am more elaborate comment, and click "Propose File Change". On the next page you get the option to "Send a pull request", which pings me that you've made changes. (note that you now forked Moa)

### 3.11.2 Software Prerequisites

Obviously, in order to build documentation you will need to install [Sphinx](#) using the call:

```
pip install Sphinx
```

For Python 2.6 users [argparse](#) is also required so call:

```
pip install argparse
```

### 3.11.3 Code

You can easily fork the repository and hack away. You can send pull requests if you want your new code merged into the central Moa repository. Contact [me](#) if you have any questions.

## 3.12 Acknowledgements

The following people have contributed to Moa by writing code, using and testing, submitting bug reports and feature requests, writing documentation or engaging into general discussion.



John McCallum, Roy Storey, Marcus Davy, Susan Thomson, Ashley Lu, Cecilia Deng, Helge Dzierzon, Yogini Idnani, William Demchick and Mark Fiers

Plant & Food Research New Zealand has provided financial support for development

**NOTE: both the software and the manual are under development. Expect things to change.**

### 3.13 How to write a template

A MOA template is made up of a `.moa` file and a `.jinja2` (or `.mk`) file.

The `.moa` file mainly contains input-output file sets and parameter options used for the bash command(s). Some of these options have default values which the user can change while constructing the job.

The `.jinja2` file includes information to structure the command(s). It is written in `jinja`, which is a templating language for python and is simple to write and easy to understand.

These files are used by the backend, currently *ruffus*, that manages file set and parameter dependencies to make pipelines and render commands to the bash prompt. Initially, *GNU make* was the backend used. It is very powerful but some of its limitations and its complexity led to including *ruffus* as an option for the backend as well.

The easiest way to write a moa template is to edit an existing template to suit your requirements. This involves understanding the parts of an existing template.

The `bwa_aln` template is used as an example below. Just as a background, the `bwa aln` command takes a FASTQ file as input and aligns it to a reference genome that was previously indexed. The output is a `.sai` file with the alignments.

The `bwa_aln.moa` file has some main components:

- *Backend*

```
backend: ruff
```

This is ‘ruff’ which means that *ruffus* is used in the python script at a lower level to read the template `.moa` and `.jinja2` file, and render the corresponding commands to the bash prompt.

- *Commands*

```
commands:
  run:
    mode: map
    help: run bwa aln
  clean:
    mode: simple
    help: Remove all job data, not the Moa job itself, note that this must be imple
```

This indicates the function names that you will later define. In the example above, there are 2 commands- `run` and `clean`, so `moa run` or `moa clean` on the command prompt in the job directory would execute these functions.

- *Filesets*

```
filesets:
  input:
    category: input
```

```
extension: fq
help: Fastq input files
glob: '*'
optional: false
type: set
output:
  category: output
  dir: .
  extension: sai
  glob: '{{ input_glob }}'
  source: input
  type: map
```

Like the name, each filesets refer to a set of files in a single directory. The `bwa_aln` template shows 2 filesets: `input` and `output`.

- *Category*: is essentially used to separate input from output.
- *Extension*: refers to the type of file(s) required or generated.
- *Glob*: searches for files with a specified pattern. Moa, by default (`glob=*`) automatically processes all files of the specified input extension in the directory specified. By specifying a `glob`, Moa will only process those files whose name pattern matches what is in the `glob`.
- *Type*: refers to the data type of the fileset or parameter.

A fileset can either be of `set` or `map` type. The type `set` refers to a simple set of files in a directory. The type `map` refers to a set of files that are linked to what their `source` value is. In the above code, the output fileset is mapped to the input fileset.

- *Dir*: the directory of the output fileset is `'.'`, which means that the output files will be placed in the current working directory.

- *Parameter category order*

```
parameter_category_order:
- ''
- input
- system
- advanced
```

- *Parameters*

```
mismatch_penalty:
  category: ''
  default: 3
  help: mismatch penalty
  optional: true
  type: integer
```

They are the variables/options that specify a command.

- *Category*:
- *Default*: is the value that is used by default if not changed by the user.
- *Optional*: specifies if it is necessary for the user to fill in a value for the variable. If `optional` is false, the user has to indicate a value for the parameter in order to execute the job.

- *Type*: specifies the data type of the variable eg. integer, string, boolean.

- *Moa\_id*

```
moa_id: bwa_aln
```

is supposed to be the same as the filename. Ideally something descriptive (eg. bwa\_aln). This is used to later link to the other template file.

The other template file is “bwa\_aln.jinja2” which is written in [jinja](#), a templating language for python. *Note that the jinja2 file name is the same as the moa file name.*

Important features of the bwa\_aln.jinja2 file are:

- The three hash’s (###) specify the start of a function and are followed by the function name. In our bwa\_aln example, we have defined 2 funtions: `run` and `clean`.

```
### run
```

- This definition is followed by a set of commands which you would want to be executed when you type `moa run` or `moa clean` in the bwa\_aln job directory. The commands in our example file look the same as what you would put in the command prompt but the values of the parameters are bought from the .moa file and hence it’s value is replaced by the parameter name.

```
bwa aln {{db}} \
-n {{edit_dist_missing_prob}} \
. \
. \
. \
{{ input }} \
-f {{ output }}
```

- It is also possible to add if-else statements or other computing blocks in accordance with the design language.

```
{% if color_space %} -c {% endif %}
```

## 3.14 Command reference

### 3.14.1 moa run

**usage:** `moa run [-h] [-v] [-bg] [--profile] [-j THREADS] [--ol OPENLAVAQUEUE] [--olx OPENLAVAEXTRA] [--oln OPENLAVAPROCS] [--oldummy] [--olm OPENLAVAHOST]`

execute the template ‘run’ command. Execution depends on the template. This command can only be exeucted from within a template.

**optional arguments:**

<b>-h, --help</b>	show this help message and exit
<b>-v, --verbose</b>	Show debugging output
<b>--bg</b>	Run moa in the background (implies -s)
<b>--profile</b>	Run the profiler
<b>-j THREADS</b>	No threads to use when running Ruffus

**--ol** Use OpenLava as actor

**--olq OPENLAVAQUEUE** The Openlava queue to submit this job to

**--olx OPENLAVAEXTRA** Extra arguments for bsub

**--oln OPENLAVAPROCS** The number of processors the jobs requires

**--oldummy** Do not execute - just create a script to run

**--olm OPENLAVAHOST** The host to use for openlava

~~~~~..\_command\_moa\_archive:

### 3.14.2 moa archive

Archive a job, or tree with jobs for later reuse.

This command stores only those files that are necessary for execution of this job, that is: templates & configuration. In & output files, and any other file are ignored. An exception to this are all files that start with 'moa. If the *name* is omitted, it is derived from the *jobid* parameter.

It is possible to run this command recursively with the *-r* parameter - in which case all (moa job containing) subdirectories are included in the archive.

**positional arguments:** name archive name (default: None)

**optional arguments:**

|                       |                                                                                                   |
|-----------------------|---------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>     | show this help message and exit                                                                   |
| <b>-v, --verbose</b>  | Show debugging output (default: False)                                                            |
| <b>--profile</b>      | Run the profiler (default: False)                                                                 |
| <b>-f, --force</b>    | Force this action (default: False)                                                                |
| <b>-s, --sync</b>     | Alternative approach to deal with sync type jobs - only include _ref directories (default: False) |
| <b>-t, --template</b> | Store this archive as a template (default: False)                                                 |

---

### 3.14.3 moa archive\_excl

Toggle a directory to be included in an moa archive.

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.4 moa archive\_incl

Toggle a directory to be included in an moa archive.

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.5 moa cp

Copy a moa job, or a tree with jobs (with -r).

moa cp copies only those files defining a job: the template files and the job configuration. Additionally, all files in the moa directory that start with *moa.* (for example *moa.description* are copied as well. Data and log files are not copied!. If used in conjunction with the -r (recursive) flag the complete tree is copied.

**positional arguments:** from copy from to copy to (default: None)

**optional arguments:**

|                        |                                                                                         |
|------------------------|-----------------------------------------------------------------------------------------|
| <b>-h, --help</b>      | show this help message and exit                                                         |
| <b>-v, --verbose</b>   | Show debugging output (default: False)                                                  |
| <b>--profile</b>       | Run the profiler (default: False)                                                       |
| <b>-r, --recursive</b> | copy recursively - including all subdirectories (default: False)                        |
| <b>-o, --overwrite</b> | if the target dir exists - overwrite (instead of copying into that dir (default: False) |

---

### 3.14.6 moa dumpTemplate

**moa template\_dump** - Show raw template information

Usage:

```
moa template_dump [TEMPLATE_NAME]
```

Show the raw template sysConf.

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.7 moa err

Show the stderr of the most recently executed moa job

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.8 moa files

Show in and output files for this job

Display a list of all files discovered (for input & prerequisite type filesets) and inferred from these for map type filesets.

**optional arguments:**

|                                         |                                                |
|-----------------------------------------|------------------------------------------------|
| <b>-h, --help</b>                       | show this help message and exit                |
| <b>-v, --verbose</b>                    | Show debugging output (default: False)         |
| <b>--profile</b>                        | Run the profiler (default: False)              |
| <b>-a, --all</b>                        | Show all filesets (default: False)             |
| <b>-n NO_FILES, --no_files NO_FILES</b> | No filesets to show (default 10) (default: 10) |

---

### 3.14.9 moa kill

Kill a running job.

This command checks if a job is running. If so - it tries to kill it by sending SIGKILL (-9) to the job.

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.10 moa list

Lists all known local templates

Print a list of all templates known to this moa installation. This includes locally installed templates as well.

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.11 moa lock

Lock a job - prevent execution

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.12 moa log

Show activity log

Shows a log of moa commands executed. Only commands with an impact on the pipeline are logged, such as *moa run* & *moa set*.

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.13 moa map

create an adhoc moa 'map' job

Moa will query the user for process, input & output files. A *map* job maps a set of input files on a set of output files, executing the *process* command for each combination. The *process* parameter is interpreted as a Jinja2 template with the input file available as `{{ input }}` and the output as `{{ output }}`.

**optional arguments:**

|                                |                                        |
|--------------------------------|----------------------------------------|
| <b>-h, --help</b>              | show this help message and exit        |
| <b>-v, --verbose</b>           | Show debugging output (default: False) |
| <b>--profile</b>               | Run the profiler (default: False)      |
| <b>-f, --force</b>             | Force this action (default: False)     |
| <b>-t TITLE, --title TITLE</b> | A title for this job (default: None)   |

---

### 3.14.14 moa map!

create an adhoc moa ‘map’ job

This command is exactly the same as *moa map* but uses the Moa local (or user) bash history instead.

**optional arguments:**

|                                |                                        |
|--------------------------------|----------------------------------------|
| <b>-h, --help</b>              | show this help message and exit        |
| <b>-v, --verbose</b>           | Show debugging output (default: False) |
| <b>--profile</b>               | Run the profiler (default: False)      |
| <b>-f, --force</b>             | Force this action (default: False)     |
| <b>-t TITLE, --title TITLE</b> | A title for this job (default: None)   |

---

### 3.14.15 moa mv

Move, rename or renumber a moa job.

**positional arguments:** from copy from to copy to (default: None)

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.16 moa new

Create a new job.

This command creates a new job with the specified template in the current directory. If the directory already contains a job it needs to be forced using ‘-f’. It is possible to define arguments for the job on the commandline using KEY=VALUE after the template. Note: do not use spaces around the ‘=’ sign. Use quotes if you need spaces in variables (KEY=‘two values’)

**positional arguments:** template name of the template to use for this moa job parameter arguments for this job, specifyas KEY=VALUE without spaces (default: None)

**optional arguments:**

|                                |                                        |
|--------------------------------|----------------------------------------|
| <b>-h, --help</b>              | show this help message and exit        |
| <b>-v, --verbose</b>           | Show debugging output (default: False) |
| <b>--profile</b>               | Run the profiler (default: False)      |
| <b>-f, --force</b>             | Force this action (default: False)     |
| <b>-t TITLE, --title TITLE</b> | mandatory job title (default: )        |

---



### 3.14.17 moa out

Show the stdout of the most recently executed moa job

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.18 moa pause

Pause a running job

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.19 moa postcommand

Execute 'postcommand'

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.20 moa precommand

Execute 'precommand'

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.21 moa raw\_commands

return a list available commands

Print a list of known Moa commands, both global, plugin defined commands as template specified ones. This command meant to be used by software interacting with Moa.

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.22 moa raw\_parameters

Print a list of all known parameters

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.23 moa reduce

Create a ‘reduce’ adhoc job.

There are a number of ways this command can be used:

```
$ moa reduce -t 'a title' -- echo 'define a command'
```

Anything after – will be the executable command. If omitted, Moa will query the user for a command.

Moa will also query the user for input & output files. An example session:

```
$ moa map -t 'something intelligent'
process:
> echo 'processing {{ input }} {{ output }}'
input:
> ../10.input/*.txt
output:
> ../*.out
```

Assuming you have a number of text files in the `../10/input/` directory, you will see, upon running:

```
processing ../10.input/test.01.txt ../test.01.out
processing ../10.input/test.02.txt ../test.02.out
processing ../10.input/test.03.txt ../test.03.out
...
```

**optional arguments:**

|                                |                                        |
|--------------------------------|----------------------------------------|
| <b>-h, --help</b>              | show this help message and exit        |
| <b>-v, --verbose</b>           | Show debugging output (default: False) |
| <b>--profile</b>               | Run the profiler (default: False)      |
| <b>-f, --force</b>             | Force this action (default: False)     |
| <b>-t TITLE, --title TITLE</b> | A title for this job (default: None)   |

---

### 3.14.24 moa reduce!

Create a ‘reduce’ adhoc job using the bash history

This command is exactly the same as moa reduce, but uses the bash history instead of the moa process history.

**optional arguments:**

|                                |                                        |
|--------------------------------|----------------------------------------|
| <b>-h, --help</b>              | show this help message and exit        |
| <b>-v, --verbose</b>           | Show debugging output (default: False) |
| <b>--profile</b>               | Run the profiler (default: False)      |
| <b>-f, --force</b>             | Force this action (default: False)     |
| <b>-t TITLE, --title TITLE</b> | A title for this job (default: None)   |

---

### 3.14.25 moa refresh

Refresh the template

Reload the template from the original repository.

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.26 moa rehash

cache a list of variables for command line completion

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.27 moa resume

Resume a running job

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.28 moa set

Set one or more variables

This command can be used in two ways. In its first form both parameter key and value are defined on the command line: *moa set KEY=VALUE*. Note that the command line will be processed by bash, which can either create complications or prove very useful. Take care to escape variables that you do not want to be expanded and use single quotes where necessary. For example, to include a space in a variable: *moa set KEY='VALUE WITH SPACES'*.

Alternative use of the set command is by just specifying the key: 'moa set PARAMETER\_NAME', in which case Moa will prompt the user enter a value - circumventing problems with bash interpretation.

Note: without -s, moa needs to be executed from within a Moa job

## System configuration

By specifying -s or *--system*, the variable is stored as a system configuration variable in the YAML formatted *~/.config/moa/config*. Please, use this with care!

The dots in the key name are interpreted as nested levels, so, running:

```
moa set -s plugins.job.completion.enabled=false
```

will result in the following section added on top of the YAML:

```
plugins:
  job:
    completion:
      enabled: false
```

Adding keys like this mixes safely with configuration information that is already present. So, setting:

```
moa set -s plugins.job.completion.something=else
```

will not remove the *enabled: false* heading under *completion:*, resulting in:

```
plugins:
  job:
    completion:
      enabled: false
      something: else
```

**positional arguments:** parameter arguments for this job, specify as KEY=VALUE without spaces

**optional arguments:**

|                      |                                                             |
|----------------------|-------------------------------------------------------------|
| <b>-h, --help</b>    | show this help message and exit                             |
| <b>-v, --verbose</b> | Show debugging output (default: False)                      |
| <b>--profile</b>     | Run the profiler (default: False)                           |
| <b>-f, --force</b>   | Force this action (default: False)                          |
| <b>-s, --system</b>  | store this a system configuration variable (default: False) |

---

### 3.14.29 moa show

Show parameters known to this job.

The command outputs three columns, parameter name, flag and value. The two flags have the following meaning:

- Origin: (l) locally defined; (d) default value; (r) recursively defined; (s) system defined; (x) extra value, not in the template; and (.) not defined.
- Private: a *p* indicates this variable to be private.
- Mandatory: a lower case *o* indicates this to be an optional variable and *M* means mandatory.

**optional arguments:**

|                      |                                          |
|----------------------|------------------------------------------|
| <b>-h, --help</b>    | show this help message and exit          |
| <b>-v, --verbose</b> | Show debugging output (default: False)   |
| <b>--profile</b>     | Run the profiler (default: False)        |
| <b>-u</b>            | show unrendered values (default: False)  |
| <b>-a</b>            | show all parameters (default: False)     |
| <b>-p</b>            | show private parameters (default: False) |

---

### 3.14.30 moa simple

Create a ‘simple’ adhoc job.

Simple meaning that no in or output files are tracked. Moa will query you for a command to execute (the *process* parameter). Note that Moa tracks a history for all ‘process’ parameters used.

**optional arguments:**

|                                |                                        |
|--------------------------------|----------------------------------------|
| <b>-h, --help</b>              | show this help message and exit        |
| <b>-v, --verbose</b>           | Show debugging output (default: False) |
| <b>--profile</b>               | Run the profiler (default: False)      |
| <b>-f, --force</b>             | Force this action (default: False)     |
| <b>-t TITLE, --title TITLE</b> | A title for this job (default: None)   |

---

### 3.14.31 moa simple!

Create a ‘simple’ adhoc job.

This command is exactly the same as *moa simple* except for the fact that Moa uses the bash history specific for the moa job or, if absent, the user bash history. This is convenient if you would like to register or reuse a command that you have already executed.

#### optional arguments:

|                                |                                        |
|--------------------------------|----------------------------------------|
| <b>-h, --help</b>              | show this help message and exit        |
| <b>-v, --verbose</b>           | Show debugging output (default: False) |
| <b>--profile</b>               | Run the profiler (default: False)      |
| <b>-f, --force</b>             | Force this action (default: False)     |
| <b>-t TITLE, --title TITLE</b> | A title for this job (default: None)   |

---

### 3.14.32 moa status

Show job status

Print a short status of the job, including configuration

#### optional arguments:

|                      |                                                                                          |
|----------------------|------------------------------------------------------------------------------------------|
| <b>-h, --help</b>    | show this help message and exit                                                          |
| <b>-v, --verbose</b> | Show debugging output (default: False)                                                   |
| <b>--profile</b>     | Run the profiler (default: False)                                                        |
| <b>-u</b>            | show unrendered values (when using inline parameters) (default: False)                   |
| <b>-R</b>            | show recursively defined parameters not specified by the local template (default: False) |
| <b>-p</b>            | show private parameters (default: False)                                                 |
| <b>-a</b>            | show all parameters (default: False)                                                     |

---

### 3.14.33 moa template

**moa template** - Print the template name of the current job

Usage:

`moa template`

#### optional arguments:

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |

---

|                  |                                   |
|------------------|-----------------------------------|
| <b>--profile</b> | Run the profiler (default: False) |
|------------------|-----------------------------------|

---

### 3.14.34 moa test

Test the job parameters

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.35 moa tree

Show a directory tree and job status

**positional arguments:** filter show only directories that match this filter (default: None)

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

-a, -all

---

### 3.14.36 moa unlock

Unlock a job - allow execution

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.37 moa unset

Remove a parameter from the configuration

Remove a configured parameter from this job. In the parameter was defined by the job template, it reverts back to the default value. If it was an ad-hoc parameter, it is lost from the configuration.

**positional arguments:** parameter parameter to unset

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.38 moa version

print moa version number

**optional arguments:**

|                      |                                        |
|----------------------|----------------------------------------|
| <b>-h, --help</b>    | show this help message and exit        |
| <b>-v, --verbose</b> | Show debugging output (default: False) |
| <b>--profile</b>     | Run the profiler (default: False)      |

---

### 3.14.39 msp

moa set process

Usage:

`msp`

this is an alias for the often used:

`moa set process`

## 3.15 Templates

Contents:

### 3.15.1 abyss\_pe

Run Abysspe



## Commands

**clean** Remove all job data

**run** Execute abysspe in paired-end mode

## Filesets

**fq\_forward** fastq input files directory - forward

**fq\_reverse** fastq input files directory - reverse

*type: map*  
*source: fq\_forward*  
*category: input*  
*optional: True*  
*pattern: \*/\*\_2.fq*

**output** soap denovo output file

*type: single*  
*category: output*  
*optional: True*  
*pattern: {}*

## Parameters

**joinpairs** number of pairs needed to consider joining two contigs

*type: integer*  
*default: 10*  
*optional: True*

**kmer** kmer size

*type: integer*  
*default: 31*  
*optional: True*

**threads** no threads to use

*type: integer*  
*default: 3*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Mon, 21 Nov 2011 12:47:16

**Modification date** Mon, 21 Nov 2011 12:47:22

### 3.15.2 abyss\_se

Run Abysspe

## Commands

**clean** Remove all job data

**run** Execute abyss se

## Filesets

**input** fastq input files directory

**output** soap denovo output file

*type: single*  
*category: output*  
*optional: True*  
*pattern: {}*

## Parameters

**kmer** kmer size

*type: integer*  
*default: 31*  
*optional: True*

**threads** no threads to use

*type: integer*  
*default: 3*  
*optional: True*

## **miscellaneous**

**Backend** ruff

**Author** Mark Fiers

**Creation date** Mon, 21 Nov 2011 12:47:16

**Modification date** Mon, 21 Nov 2011 12:47:22

### **3.15.3 archroot**

#### **Helper script for a root archive**

Helper script for the root of an archive template

## **Commands**

**run** *no help defined*

## **Parameters**

**moa\_archive\_parameters** space separated list of parameters to set for this template

*type: string*  
*default: {}*  
*optional: False*

## **miscellaneous**

**Backend** ruff

**Author** Mark Fiers

**Creation date** Tue, 17 Apr 2012 10:21:31

**Modification date** Tue, 17 Apr 2012 10:21:25

### **3.15.4 autohagfish**

#### **Automatically run bowtie & hagfish combined**

Run the preparatory steps for hagfish

## Commands

**clean** remove all Hagfish files

**finish** finish up - find gaps - combine plots - create a report

**run** Run hagfish

## Filesets

**fasta** fasta sequence of the reference

*type: single*  
*category: prerequisite*  
*optional: False*  
*pattern: {}*

**fw\_fq** forward fq input

**outbase** basename for output files

*type: map*  
*source: fw\_fq*  
*category: output*  
*optional: True*  
*pattern: ./\**

**rev\_fq** reverse fq input

*type: map*  
*source: fw\_fq*  
*category: input*  
*optional: True*  
*pattern: \*/\*2.fq*

## Parameters

**max\_ok** Maximal acceptable insert size for an aligned pair. If omitted, hagfish will make an estimate

*type: int*  
*default: 0*  
*optional: True*

**min\_ok** Minimal acceptable insert size for an aligned pair. If omitted, hagfish will make an estimate

*type: int*  
*default: 0*  
*optional: True*

**threads** no threads to use

*type: integer*  
*default: 8*  
*optional: True*

### miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Tue Mar 29 16:34:19 2011

**Modification date** Thu, 19 May 2011 20:49:04 +1200

### 3.15.5 bamextract

#### **bamextract**

Extract a region from a BAM file

#### **Commands**

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** Extract a region from a BAM file

#### **Filesets**

**bam** BAM input

*type: single*  
*category: input*  
*optional: False*  
*pattern: {}*

**regions** List with regions to extract (id seqid start stop)

*type: single*  
*category: input*  
*optional: False*  
*pattern: {}*

**vcf** optional VCF input

*type: single*  
*category: input*  
*optional: True*  
*pattern: {}*

## Parameters

**flank** flanking region to extract

*type: integer*  
*default: 100*  
*optional: {}*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Wed Nov 10 07:56:48 2010

### 3.15.6 bdbb

#### Bidirectional best BLAST hit

Discover the bidirectional best blast hit between two sets of sequences

## Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** generate a list of bidirectional best blast hits between two databases of sequences

## Filesets

**input\_a** First multi fasta input set

*type: single*  
*category: input*  
*optional: False*  
*pattern: \*/\*.fasta*

**input\_b** Second multi fasta input set

*type: single*  
*category: input*  
*optional: False*  
*pattern: \*/\*.fasta*

**output** List of bidirectional best blasts hits

*type: map*  
*source: input\_a*  
*category: output*  
*optional: True*  
*pattern: \*/\*.list*

## Parameters

**eval** e value cutoff

*type: float*  
*default: 1e-10*  
*optional: True*

**extract** Extract the identified sequences from the input fasta files

*type: boolean*  
*default: False*  
*optional: True*

**nothreads** Threads to run blast with with

*type: integer*  
*default: 4*  
*optional: True*

**protein** Is this a protein set

*type: boolean*  
*default: False*  
*optional: True*

**tblastx** If this is a nucleotide set, use tblastx?? (otherwise use blastn)

*type: boolean*  
*default: F*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** unknown

### 3.15.7 bfast\_aln

Generate bam format alignments using bfast

## Commands

**clean** Remove all job data, not the Moa job itself

**run** run bfast match, localalign, postprocess commands

## Filesets

**fa\_input** fasta input file

**fq\_input** fastq input files

**output\_aln**

*type: map*  
*source: fq\_input*  
*category: output*



*optional: {}*  
*pattern: ./\*.aln*

**output\_bam**

*type: map*  
*source: fq\_input*  
*category: output*  
*optional: {}*  
*pattern: ./\*.bam*

**Parameters**

**algorithm\_colour\_space** true -> colour space, false -> NT space

*type: boolean*  
*default: False*  
*optional: True*

**avg\_mism\_qual** Specifies the average mismatch quality

*type: integer*  
*default: 10*  
*optional: True*

**extra\_params\_localalign** Any extra parameters for the localalign command

*type: string*  
*default: ""*  
*optional: True*

**extra\_params\_match** Any extra parameters for the match command

*type: string*  
*default: ""*  
*optional: True*

**extra\_params\_postprocess** Any extra parameters for the postprocess command

*type: string*

*default: ""*  
*optional: True*

**min\_mapping\_qual** Specifies to remove low mapping quality alignments

*type: integer*  
*default: -2147483648*  
*optional: True*

**min\_norm\_score** Specifies to remove low (alignment) scoring alignments

*type: integer*  
*default: -2147483648*  
*optional: True*

**output\_format** 0 - BAF, 1 - SAM

*type: integer*  
*default: 1*  
*optional: True*

**paired\_opp\_strands** Specifies that paired reads are on opposite strands

*type: boolean*  
*default: False*  
*optional: True*

**pairing\_std\_dev** Specifies the pairing distance standard deviation to examine when recuing

*type: float*  
*default: 2.0*  
*optional: True*

**print\_params** print program parameters

*type: boolean*  
*default: False*

*optional: True*

**thread\_num** Specifies the number of threads to use

*type: integer*

*default: 1*

*optional: True*

**timing\_information** specifies output timing information

*type: boolean*

*default: True*

*optional: True*

**ungapped\_aln** Do ungapped local alignment

*type: boolean*

*default: False*

*optional: True*

**ungapped\_pairing\_rescue** Specifies that ungapped pairing rescue should be performed

*type: boolean*

*default: False*

*optional: True*

**unpaired\_reads** True value specifies that pairing should not be performed

*type: boolean*

*default: False*

*optional: True*

**usage\_summary** Display usage summary (help)

*type: boolean*

*default: False*

*optional: True*

**which\_strand** 0 - consider both strands, 1 - forwards strand only, 2 - reverse strand only

*type: integer*  
*default: 0*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Yogini Idnani, Mark Fiers

**Creation date** Wed Feb 15 10:06:48 2011

**Modification date** unknown

### 3.15.8 bfast\_db

Generate db index files for aligning reads with bfast

## Commands

**clean** Remove all job data, not the Moa job itself

**run** run bfast fasta2brg and index commands

## Filesets

**fa\_input** fasta input file

## Parameters

**algorithm\_colour\_space** true -> colour space, false -> NT space

*type: boolean*  
*default: False*  
*optional: True*

**depth** The depth of the splitting(d). The index will be split into 4<sup>d</sup> parts.

*type: integer*  
*default: 0*  
*optional: True*

**extra\_params** Any extra parameters

*type: string*  
*default: ""*  
*optional: True*

**hash\_width** The hash width for the index (recommended from manual = 14)

*type: integer*  
*default: {}*  
*optional: False*

**index\_num** Specifies this is the ith index you are creating

*type: integer*  
*default: 1*  
*optional: True*

**mask** The mask or spaced seed to use.

*type: string*  
*default: {}*  
*optional: False*

**print\_params** print program parameters

*type: boolean*  
*default: False*  
*optional: True*

**thread\_num** Specifies the number of threads to use

*type: integer*  
*default: 1*  
*optional: True*

**timing\_information** specifies output timing information

*type: boolean*

*default: True*

*optional: True*

**usage\_summary** Display usage summary (help)

*type: boolean*

*default: False*

*optional: True*

## miscellaneous

**Backend** ruff

**Author** Yogini Idnani, Mark Fiers

**Creation date** Wed Feb 15 10:06:48 2011

**Modification date** unknown

## 3.15.9 blast

### Basic Local Alignment Tool

Wraps BLAST [[Alt90]], probably the most popular similarity search tool in bioinformatics.

## Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**report** Generate a text BLAST report.

**run** Running BLAST takes an input directory, determines what sequences are present and executes BLAST on each of these. Moa BLAST is configured to create XML output (as opposed to the standard text based output) in the out directory. The output XML is subsequently converted to GFF3 by the custom blast2gff script (using BioPython). Additionally, a simple text report is created.

## Filesets

**db** Blast database

*type: single*

*category: prerequisite*

*optional: False*

*pattern: \*/\**

**input** Directory with the input files for BLAST, in Fasta format

**outgff** GFF output files

*type: map*  
*source: input*  
*category: output*  
*optional: True*  
*pattern: gff/\*.gff*

**output** XML blast output files

*type: map*  
*source: input*  
*category: output*  
*optional: True*  
*pattern: out/\*.out*

## Parameters

**eval** e value cutoff

*type: float*  
*default: 1e-10*  
*optional: True*

**gff\_blasthit** (T,\*\*F\*\*) - export an extra blasthit feature to the created gff, grouping all hsp (match) features.

*type: set*  
*default: F*  
*optional: True*

**gff\_source** source field to use in the gff

*type: string*  
*default: BLAST*  
*optional: True*

**nohits** number of hits to report

*type: integer*  
*default: 50*  
*optional: True*

**nothreads** threads to run blast with (note the overlap with the Make -j parameter)

*type: integer*  
*default: 2*  
*optional: True*

**program** blast program to use (default: blastn)

*type: set*  
*default: blastn*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Wed Nov 10 07:56:48 2010

### 3.15.10 blastdb

#### Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** Takes either a set of fasta files or a single multi-fasta input file and creates a BLAST database.

#### Filesets

##### dbname

*type: map*  
*source: input*  
*category: output*  
*optional: {}*  
*pattern: ./db*

**input** The file with all input FASTA sequences for the blastdb.



*type: single*  
*category: input*  
*optional: False*  
*pattern: \*/\*.fasta*

## Parameters

**protein** Protein database? (T)rue or not (F)alse (default: F)

*type: set*  
*default: F*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Tue, 03 Jan 2012 15:00:23

### 3.15.11 bowtie

#### Bowtie

Run BOWTIE on an set of input files (query) vs a database index.

## Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template

**run** *no help defined*

## Filesets

**input** Fasta/fastq input files for bowtie

**output** Output files

*type: map*  
*source: input*  
*category: output*  
*optional: {}*  
*pattern: ./\*.bam*

## Parameters

**db** The (basename of the) bowtie database to use.

*type: string*  
*default: {}*  
*optional: False*

**extra\_params** extra parameters to feed bowtie

*type: string*  
*default: ""*  
*optional: True*

**input\_format** Format of the input files

*type: set*  
*default: fastq*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Wed Nov 10 07:56:48 2010

### 3.15.12 bowtie\_pe

Run BOWTIE on an set of input files (query) vs a database index.

## Commands

**clean** Remove all job data, not the Moa job itself

**finish** finish up

**report** Create a report on the results

**run** Execute soapdenovo in paired-end mode

## Filesets

**db** The (basename of the) bowtie database to use.

*type: single*  
*category: prerequisite*  
*optional: False*  
*pattern: ../20.bowtiedb/db*

**fq\_forward\_input** Fastq input files - forward

**fq\_reverse\_input** Fastq input files - reverse

*type: map*  
*source: fq\_forward\_input*  
*category: input*  
*optional: True*  
*pattern: \*/\*\_2.fq*

**output** Bam output file

*type: map*  
*source: fq\_forward\_input*  
*category: output*  
*optional: {}*  
*pattern: \*/\*.bam*

## Parameters

**extra\_params** extra parameters to feed to bowtie

*type: string*  
*default: ""*  
*optional: True*

**input\_format** Format of the input files

*type: set*  
*default: fastq*  
*optional: True*

**lots\_of\_data** Keep unmapped reads, unsorted BAM - takes up a lot of space!

*type: boolean*

*default: False*

*optional: True*

**max\_insertsize** Maximum allowed insertsize

*type: integer*

*default: 250*

*optional: True*

**min\_insertsize** Minimum allowed insertsize

*type: integer*

*default: 1*

*optional: True*

**orientation** orientation of the reads, allowed values are fr, rf, ff

*type: {}*

*default: fr*

*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Wed Nov 10 07:56:48 2010

### 3.15.13 bowtie\_se

Run BOWTIE on an set of input files (query) vs a database index.

## Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template

**run** *no help defined*

## Filesets

**fq\_input** fastq input files directory

**output** Bam output file

*type: map*  
*source: fq\_input*  
*category: output*  
*optional: {}*  
*pattern: /\*.bam*

## Parameters

**ebwt\_base** The (basename of the) bowtie database to use.

*type: string*  
*default: {}*  
*optional: False*

**extra\_params** extra parameters to feed to bowtie

*type: string*  
*default: ""*  
*optional: True*

**input\_format** Format of the input files

*type: set*  
*default: fastq*  
*optional: True*

**output\_format** Format of the output file

*type: set*  
*default: bam*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Yogini Idnani, Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Wed Nov 10 07:56:48 2010

### 3.15.14 bowtiedb

Builds a bowtie index from a reference sequence

#### Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** Create the bowtie database

#### Filesets

**input** Input fasta file for the bowtie database

**output** database name to create

*type: single*

*category: output*

*optional: {}*

*pattern: db*

#### Parameters

**extra\_params** any option parameters

*type: string*

*default: ""*

*optional: True*

**title**

*type: {}*

*default: Bowtie index builder*

*optional: {}*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Wed Dec 09 07:56:48 2010

### 3.15.15 bwa\_aln

Use BWA to align a set of fastq reads against a db

#### Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** run bwa aln

#### Filesets

**input** Fastq input files

**output**

*type: map*  
*source: input*  
*category: output*  
*optional: {}*  
*pattern: ./\*.sai*

#### Parameters

**best\_hits\_stop** stop searching when there are >INT equally best hits

*type: integer*  
*default: {}*  
*optional: True*

**color\_space** input sequences are in the color space

*type: boolean*  
*default: False*  
*optional: True*

**db** bwa database to align against

*type: string*  
*default: {}*  
*optional: False*

**edit\_dist\_missing\_prob** max

*type: float*  
*default: {}*  
*optional: True*

**gap\_ext\_max**

*type: integer*  
*default: {}*  
*optional: True*

**gap\_ext\_penalty** gap extension penalty

*type: integer*  
*default: {}*  
*optional: True*

**gap\_open\_penalty** gap open penalty

*type: integer*  
*default: {}*  
*optional: True*

**gap\_opens\_max** maximum number or fraction of gap opens

*type: integer*  
*default: {}*  
*optional: True*

**log\_gap\_penalty\_del** log-scaled gap penalty for long deletions

*type: boolean*  
*default: {}*  
*optional: True*



**max\_ext\_long\_del** maximum occurrences for extending a long deletion

*type: integer*  
*default: {}*  
*optional: True*

**max\_queue\_entry** maximum entries in the queue

*type: integer*  
*default: {}*  
*optional: True*

**mismatch\_penalty** mismatch penalty

*type: integer*  
*default: {}*  
*optional: True*

**no\_indel\_from\_ends** do not put an indel within INT bp towards the ends

*type: integer*  
*default: {}*  
*optional: True*

**non\_iterative** non-iterative mode search for all n-difference hits (slow)

*type: boolean*  
*default: False*  
*optional: True*

**quality\_step** quality threshold for read trimming down to 35bp

*type: integer*  
*default: {}*  
*optional: True*

**seed\_len** Seed length

*type: integer*  
*default: {}*  
*optional: True*

**seed\_max\_diff** Maximum differences in the seed

*type: integer*  
*default: {}*  
*optional: True*

**thread\_num** number of threads

*type: integer*  
*default: {}*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers, Yogini Idnani

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** unknown

### 3.15.16 bwa\_index

#### Bwa index builder

Builds a bwa index from a reference sequence

#### Commands

**clean** Remove all job data

**run** Create the index

#### Filesets

**input** Input fasta file for the bowtie database

*type: single*  
*category: input*  
*optional: False*

*pattern: \*/\*.fasta*

## Parameters

**algorithm** Algorithm for constructing BWT index. Available options are ‘is’ and ‘bwtsv’

*type: string*

*default: is*

*optional: True*

**color\_space** input sequences are in the color space

*type: boolean*

*default: False*

*optional: True*

**prefix** Name of the bwa index to create

*type: string*

*default: db*

*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers, Yogini Idnani

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Wed Nov 10 07:56:48 2010

### 3.15.17 bwa\_sampe

Generate alignments in SAM format given paired end reads

## Commands

**clean** Remove all job data, not the Moa job itself

**run** run bwa sampe

## Filesets

**fq\_forward\_input** fastq input files directory - forward

**fq\_reverse\_input** fastq input files directory - reverse

*type: map*  
*source: fq\_forward\_input*  
*category: input*  
*optional: True*  
*pattern: \*/\*\_2.fq*

**output\_bam**

*type: map*  
*source: fq\_forward\_input*  
*category: output*  
*optional: {}*  
*pattern: ./\*.bam*

**sai\_forward\_input** sai input files - forward

*type: map*  
*source: fq\_forward\_input*  
*category: input*  
*optional: False*  
*pattern: \*/\*\_1.sai*

**sai\_reverse\_input** sai input files - reverse files

*type: map*  
*source: sai\_forward\_input*  
*category: input*  
*optional: True*  
*pattern: \*/\*\_2.sai*

## Parameters

**db** bwa database to align against

*type: string*  
*default: {}*  
*optional: False*

**disable\_insert\_size** disable insert size estimate (force -s)

*type: boolean*

*default: False*

*optional: True*

**disable\_SW** disable Smith-Waterman for the unmapped mate

*type: boolean*

*default: False*

*optional: True*

**lots\_of\_data** store unmapped reads - takes up a lot of space!

*type: boolean*

*default: False*

*optional: True*

**max\_aln\_out** maximum hits to output for paired reads

*type: integer*

*default: 3*

*optional: True*

**max\_insert\_size** maximum insert size

*type: integer*

*default: 500*

*optional: True*

**max\_occ\_read** maximum occurrences for one end

*type: integer*

*default: {}*

*optional: True*

**max\_out\_discordant\_pairs** maximum hits to output for discordant pairs

*type: integer*  
*default: {}*  
*optional: True*

**preload\_index** preload index into memory (for base-space reads only)

*type: boolean*  
*default: False*  
*optional: True*

**prior\_chimeric\_rate** prior of chimeric rate (lower bound)

*type: integer*  
*default: {}*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Yogini Idnani, Mark Fiers

**Creation date** Wed Nov 25 17:06:48 2010

**Modification date** unknown

### 3.15.18 bwa\_samse

Generate alignments in SAM format given single end reads, using both ‘bwa samse’.

## Commands

**clean** Remove all job data, not the Moa job itself

**run** run bwa samse

## Filesets

**fq\_input** fastq input file

**output\_bam** output bam file

*type: map*  
*source: fq\_input*  
*category: output*

*optional: {}*  
*pattern: ./\*.bam*

**sai\_input** sai input directory - filenames must correspond to the fastq input files

*type: map*  
*source: fq\_input*  
*category: input*  
*optional: False*  
*pattern: \*/\*.sai*

## Parameters

**db** bwa database to align against

*type: string*  
*default: ""*  
*optional: False*

**max\_aln\_out** Maximum number of alignments to output in the XA tag for reads paired properly

*type: integer*  
*default: 3*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Yogini Idnani, Mark Fiers

**Creation date** Wed Nov 25 17:06:48 2010

**Modification date** unknown

### 3.15.19 cdsmatrix

#### CdsMatrix

Predicts (prokaryotic) using glimmer3.

## Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** Generate a matrix of CDS's

## Filesets

**input** Directory with the cds files for Glimmer3

**output** Output blast files

*type: map*  
*source: input*  
*category: output*  
*optional: True*  
*pattern: ./\*.out*

**reference** reference multi fasta file

*type: single*  
*category: prerequisite*  
*optional: {}*  
*pattern: \*/\*.fasta*

**table** table files

*type: map*  
*source: input*  
*category: output*  
*optional: True*  
*pattern: ./\*.tab*

## Parameters

**cutoff** score cutoff value - disregards hits below this score

*type: {}*  
*default: 100*  
*optional: True*



## **miscellaneous**

**Backend** ruff

**Author** Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Thu, 21 Jul 2011 20:31:10 +1200

### **3.15.20 empty**

**empty**

Do nothing...

## **Commands**

## **Parameters**

## **miscellaneous**

**Backend** ruff

**Author** Mark Fiers

**Creation date** Mon Apr 04 16:02:58 2011

**Modification date** Mon Apr 04 16:03:18 2011

### **3.15.21 fastainfo**

**gather information on a set of fasta files**

gather info on a set of input files

## **Commands**

**finish** create a report

**run** generate info on each of the input sequences

## **Filesets**

**input** “fastainfo” input files

**output** “fastainfo” raw output files

*type: map*

*source: input*

*category: output*

*optional: True*  
*pattern: stats/\*.out*

**stats** “fastainfo” collect stat files

*type: map*  
*source: input*  
*category: output*  
*optional: True*  
*pattern: stats/\*.stat*

## Parameters

### miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Mon, 11 Jul 2011 15:15:20

**Modification date** Mon, 11 Jul 2011 15:15:12

### 3.15.22 fastqc

#### Run FastQC for fastq QC

Run FastQC on a set a fastq files - quality assessment

## Commands

**finish** Run Fastqc

**finish** delegates execution to: **report**

**report** Generate a simple fastqc report

**run** *no help defined*

## Filesets

**input** fastqc input files'

**touch** touch files - track if a file has been processed - do not touch this unless you know what you're doing.

*type: map*  
*source: input*

*category: output*  
*optional: True*  
*pattern: /\*.touch*

## Parameters

**output\_dir** output directory for the fastQC report

*type: dir*  
*default: .*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Thu, 28 Apr 2011 09:27:17 +1200

**Modification date** Thu, 28 Apr 2011 14:19:04 +1200

### 3.15.23 fastx\_clipper

run fastx\_clipper

## Commands

**clean** Remove all job data, not the Moa job itself

**run** run fastx\_clipper

## Filesets

**input** fastq input files directory

**output**

*type: map*  
*source: input*  
*category: output*  
*optional: {}*  
*pattern: /\*.fq*

## Parameters

**adaptor** ADAPTER string. default is CCTTAAGG (dummy adapter).

*type: string*

*default: CCTTAAGG*

*optional: True*

**adaptor\_and\_bases** Keep the adapter and N bases after it.

*type: integer*

*default: 0*

*optional: True*

**compress\_output** Compress output with GZIP.

*type: boolean*

*default: False*

*optional: True*

**debug\_output** DEBUG output.

*type: boolean*

*default: False*

*optional: True*

**help** help screen

*type: boolean*

*default: False*

*optional: True*

**keep\_unknown\_nuc\_seq** keep sequences with unknown (N) nucleotides. default is to discard such sequences.

*type: boolean*

*default: False*

*optional: True*

**out\_adaptor\_only\_seq** Report Adapter-Only sequences.

*type: boolean*

*default: False*

*optional: True*

**rm\_clipped\_seq** Discard clipped sequences (i.e. - keep only sequences which did not contained the adaptor).

*type: boolean*

*default: False*

*optional: True*

**rm\_non\_clipped\_seq** Discard non-clipped sequences (i.e. - keep only sequences which contained the adaptor).

*type: boolean*

*default: False*

*optional: True*

**rm\_short\_seq** discard sequences shorter than N nucleotides. default is 5.

*type: integer*

*default: 5*

*optional: True*

**verbose** Verbose - report number of sequences. If [-o] is specified, report will be printed to STDOUT. If [-o] is not specified (and output goes to STDOUT), report will be printed to STDERR.

*type: boolean*

*default: False*

*optional: True*

## **miscellaneous**

**Backend** ruff

**Author** Mark Fiers, Yogini Idnani

**Creation date** Wed Dec 06 17:06:48 2010

**Modification date** unknown

### 3.15.24 fastx\_qual\_stats

run fastx\_quality\_stats, fastq\_quality\_boxplot\_graph.sh and  
fastx\_nucleotide\_distribution\_graph.sh

#### Commands

**clean** Remove all job data, not the Moa job itself

**run** run fastx\_quality\_stats, fastq\_quality\_boxplot\_graph.sh and fastx\_nucleotide\_distribution\_graph.sh

#### Filesets

##### boxplot\_output

*type: map*  
*source: input*  
*category: output*  
*optional: {}*  
*pattern: /\*.png*

**input** fastq input files directory

##### nuc\_distr\_output

*type: map*  
*source: input*  
*category: output*  
*optional: {}*  
*pattern: /\*.png*

##### qual\_output

*type: map*  
*source: input*  
*category: output*  
*optional: {}*  
*pattern: /\*.txt*

#### Parameters

**gen\_postScript\_file** Generate PostScript (.PS) file. Default is PNG image.

*type: boolean*  
*default: False*  
*optional: True*

**graph\_title** Title - will be plotted on the graph.

*type: string*  
*default: {{ input\_glob }}*  
*optional: True*

**help** help screen

*type: boolean*  
*default: False*  
*optional: True*

**new\_out\_format** New output format (with more information per nucleotide/cycle)

*type: boolean*  
*default: False*  
*optional: True*

## **miscellaneous**

**Backend** ruff

**Author** Mark Fiers, Yogini Idnani

**Creation date** Wed Dec 03 17:06:48 2010

**Modification date** unknown

### **3.15.25 filterwgs\_pair**

**Execute a “map22” ad-hoc analysis - two input files, two output files**

Filter raw WGS data

## **Commands**

**run** Filter WGS data

## **Filesets**

**input1** forward input fastq

**input2** reverse input fastq

*type: map*  
*source: input1*  
*category: input*  
*optional: False*  
*pattern: \*/\**

**output1** forward output fastq

*type: map*  
*source: input1*  
*category: output*  
*optional: True*  
*pattern: ./\**

**output2** reverse output fastq

*type: map*  
*source: input1*  
*category: output*  
*optional: True*  
*pattern: ./\**

## Parameters

**adapters** Fasta file with the adapter sequences to trim

*type: file*  
*default: {}*  
*optional: False*

**minlen** Minimum remaining sequence length

*type: int*  
*default: 50*  
*optional: True*

**qual** quality threshold causing trimming

*type: int*



*default: 13*  
*optional: True*

**title**

*type: {}*  
*default: Filter paired fastq files using fastq-mcf*  
*optional: {}*

**miscellaneous**

**Backend** ruff

**Author** Mark Fiers

**Creation date** Tue Mar 29 16:34:19 2011

**Modification date** Mon, 13 Feb 2012 09:16:36 +1300

**3.15.26 genemarks****geneMarkS**

predict genes using geneMarkS

**Commands**

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** *no help defined*

**Filesets**

**input** Directory with the input files for Genemarks

**Parameters**

**gff\_source** source field to use in the gff. Defaults to “geneMarkS”

*type: string*  
*default: genemarkS*  
*optional: True*

**matrix** the matrix to use

*type: file*

*default: ""*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author**

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Wed Nov 10 07:56:48 2010

### 3.15.27 glimmer3

#### Glimmer3

Predicts (prokaryotic) using glimmer3.

#### Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** Glimmer3 is a open reading frame discovery program from the EMBOSS [[emboss]] package. It takes a set of input sequences and predicts all open reading frames. Additionally, this template converts the default output (predicted protein sequences) to GFF3.

#### Filesets

**cds** CDS output files from glimmer3

*type: map*  
*source: input*  
*category: output*  
*optional: True*  
*pattern: cds/\*.fasta*

**gff** GFF output files from glimmer3

*type: map*  
*source: input*  
*category: output*  
*optional: True*  
*pattern: gff/\*.gff*

**input** Directory with the input files for Glimmer3

**output** Raw output files from glimmer3

*type: map*  
*source: input*  
*category: output*  
*optional: True*  
*pattern: out/\*.g3*

**pep** peptide output files from glimmer3

*type: map*  
*source: input*  
*category: output*  
*optional: True*  
*pattern: pep/\*.fasta*

## Parameters

**gene\_len** Minimum gene length (glimmer3 -g/-gene\_len)

*type: integer*  
*default: 110*  
*optional: True*

**gff\_source** source field to use in the gff. Defaults to “glimmer3”

*type: string*  
*default: glimmer3*  
*optional: True*

**max\_overlap** Maximum overlap, see the glimmer documentation for the -o or -max\_olap parameter

*type: integer*  
*default: 50*  
*optional: True*

**stop\_codons** stop codons

*type: {}*  
*default: tag,tga,taa,nnn,tnn,ann,gnn,cnn*  
*optional: True*

**threshold** threshold for calling a gene a gene (glimmer3 -t)

*type: integer*  
*default: 30*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Wed Nov 10 07:56:48 2010

## 3.15.28 gmap

### Gmap

Run GMAP on an set of input files (query) vs a database index.

## Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** *no help defined*

## Filesets

### align

*type: map*  
*source: input*  
*category: output*  
*optional: {}*  
*pattern: ./align/\*.align*

### genepred

*type: map*  
*source: input*  
*category: output*  
*optional: {}*

*pattern: ./genepred/\*.genepred*

**gff**

*type: map*

*source: input*

*category: output*

*optional: {}*

*pattern: ./gff/\*.gff*

**gff\_invert**

*type: map*

*source: input*

*category: output*

*optional: {}*

*pattern: ./gff/\*.invert.gff*

**input** Sequences to map**raw**

*type: map*

*source: input*

*category: output*

*optional: {}*

*pattern: ./raw/\*.raw*

**Parameters****db** Gmap db

*type: file*

*default: ""*

*optional: False*

**extra\_parameters** extra parameters to feed to gmap

*type: string*

*default: ""*

*optional: True*

**gff\_source** Source field to use in the output GFF

*type: string*

*default: gmap*

*optional: True*

**invert\_gff** Invert the GFF (T/F)

*type: set*

*default: T*

*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Wed Nov 10 07:56:48 2010

## 3.15.29 gmapdb

### gmapdb index builder

Builds gmapdb index from a reference sequence

## Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** *no help defined*

## Filesets

**input** The reference sequence to build a gmap database with.

*type: single*

*category: input*

*optional: False*

*pattern: \*/\*.fasta*

## Parameters

**name** Name of the gmap index to create

*type: string*

*default: gmapdb*

*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Wed Nov 10 07:56:48 2010

### 3.15.30 hagfish

#### Run hagfish\_extract & hagfish\_combine

Run the preparatory steps for hagfish

#### Commands

**clean** remove all Hagfish files

**finish** finish up - find gaps - combine plots - create a report

**run** Run hagfish

#### Filesets

**fasta** fasta sequence of the reference

*type: single*

*category: prerequisite*

*optional: False*

*pattern: {}*

**input** “hagfish” input files

**output** “hagfish” touch files - track what files are done - please do not touch this!

*type: map*

*source: input*

*category: output*

*optional: True*

*pattern: ./touch/\*.touch*

## Parameters

**circosbinsize** Binsize for generating circos formatted histograms

*type: int*  
*default: {}*  
*optional: True*

**max\_ok** Maximal acceptable insert size for an aligned pair. If omitted, hagfish will make an estimate

*type: int*  
*default: 0*  
*optional: True*

**min\_ok** Minimal acceptable insert size for an aligned pair. If omitted, hagfish will make an estimate

*type: int*  
*default: 0*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Tue Mar 29 16:34:19 2011

**Modification date** Thu, 19 May 2011 20:49:04 +1200

### 3.15.31 kanga

use kanga to align short reads to a reference genome

## Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** run kanga

## Filesets

**input\_fasta** Fasta input file

**output** output files



*type: map*  
*source: rds\_input*  
*category: output*  
*optional: True*  
*pattern: ./\*.sam*

**output\_bam** output files

*type: map*  
*source: rds\_input*  
*category: output*  
*optional: True*  
*pattern: ./\*.bam*

**output\_log** output log file

*type: map*  
*source: rds\_input*  
*category: output*  
*optional: {}*  
*pattern: ./\*.log.txt*

**rds\_input** rds (preprocessed) input files

**sfx\_input** sfx array lookup file

## Parameters

**color\_space** process for colorspace (SOLiD)

*type: boolean*  
*default: False*  
*optional: True*

**extra\_params** any extra parameters

*type: string*  
*default: ""*  
*optional: True*

**help** print this help and exit

*type: boolean*  
*default: False*  
*optional: True*

**max\_Ns** maximum number of intermediate N's in reads before treating read as unalignable

*type: integer*  
*default: 1*  
*optional: True*

**max\_pair\_len** accept paired end alignments with apparent length of at most this

*type: integer*  
*default: 300*  
*optional: True*

**min\_pair\_len** accept paired end alignments with apparent length of at least this

*type: integer*  
*default: 100*  
*optional: True*

**no\_multireads** do not accept multiple reads aligning to the same loci

*type: boolean*  
*default: False*  
*optional: True*

**out\_format** 0 - CSV loci only, 1 - CSV loci + match sequence, 2 - CSV loci + read sequence, 3 - CSV loci + read + match sequence, 4 - UCSC BED, 5 - SAM format

*type: integer*  
*default: 0*  
*optional: True*

**pe\_mode** 0 - none, 1 - paired ends with recover orphan ends, 2 - paired end no orphan recovery

*type: integer*  
*default: 0*  
*optional: True*

**quality** fastq quality scoring- 0 - sanger, 1m - Illumina 1.3+, 2 - Solexa < 1.3, 3 - Ignore quality

*type: integer*  
*default: 3*  
*optional: True*

**thread\_num** number of processing threads (0 sets threads to number of CPU cores)

*type: integer*  
*default: 0*  
*optional: True*

**trim3** trim this number of bases from 3' end of reads when loading raw reads

*type: integer*  
*default: 0*  
*optional: True*

**trim5** trim this number of bases from 5' end of reads when loading raw reads

*type: integer*  
*default: 0*  
*optional: True*

**version** print version information and exit

*type: boolean*  
*default: False*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers, Yogini Idnani

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** unknown

### 3.15.32 kangar\_pe

use kangar to pre process raw fq reads

## Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** run kangar

## Filesets

**fq\_forward\_input** fastq input files - forward - containing the 5' end

**fq\_reverse\_input** fastq input files directory - reverse - containing the 3' end

*type: map*  
*source: fq\_forward\_input*  
*category: input*  
*optional: True*  
*pattern: \*/\*\_2.fq*

**output\_log** output log file

*type: map*  
*source: fq\_forward\_input*  
*category: output*  
*optional: {}*  
*pattern: ./\*.log.txt*

**rds\_output** output rds file

*type: map*  
*source: fq\_forward\_input*  
*category: output*

*optional: True*  
*pattern: /\*.rds*

## Parameters

**extra\_params** any extra parameters

*type: string*  
*default: ''*  
*optional: True*

**help** print this help and exit

*type: boolean*  
*default: False*  
*optional: True*

**mode** processing mode 0 - single end create, 1 - paired end create, 2 - output statistics 3 - dump as fasta

*type: integer*  
*default: 0*  
*optional: True*

**quality** fastq quality scoring- 0 - sanger, 1m - Illumina 1.3+, 2 - Solexa < 1.3, 3 - Ignore quality

*type: integer*  
*default: 3*  
*optional: True*

**reads\_num** limit number of reads (or dumps) in each input file to this many, 0 if no limit

*type: integer*  
*default: 0*  
*optional: True*

**rm\_duplicates** remove duplicate reads retaining only one

*type: boolean*

*default: False*  
*optional: True*

**trim3** trim this number of bases from 3' end of sequence

*type: integer*  
*default: 0*  
*optional: True*

**trim5** trim this number of bases from 5' end of sequence

*type: integer*  
*default: 0*  
*optional: True*

**version** print version information and exit

*type: boolean*  
*default: False*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers, Yogini Idnani

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** unknown

### 3.15.33 kangar\_se

use kangar to pre process raw fq single end reads

## Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** run kangar

## Filesets

**fq\_input** fastq input files - forward - containing the 5' end

**output\_log** output log file

*type: map*  
*source: fq\_input*  
*category: output*  
*optional: {}*  
*pattern: /\*.log.txt*

**rds\_output** output rds file

*type: map*  
*source: fq\_input*  
*category: output*  
*optional: True*  
*pattern: /\*.rds*

## Parameters

**extra\_params** any extra parameters

*type: string*  
*default: ""*  
*optional: True*

**help** print this help and exit

*type: boolean*  
*default: False*  
*optional: True*

**mode** processing mode 0 - single end create, 1 - paired end create, 2 - output statistics 3 - dump as fasta

*type: integer*  
*default: 0*  
*optional: True*

**quality** fastq quality scoring- 0 - sanger, 1m - Illumina 1.3+, 2 - Solexa < 1.3, 3 - Ignore quality

*type: integer*  
*default: 3*  
*optional: True*

**reads\_num** limit number of reads (or dumps) in each input file to this many, 0 if no limit

*type: integer*  
*default: 0*  
*optional: True*

**rm\_duplicates** remove duplicate reads retaining only one

*type: boolean*  
*default: False*  
*optional: True*

**trim3** trim this number of bases from 3' end of sequence

*type: integer*  
*default: 0*  
*optional: True*

**trim5** trim this number of bases from 5' end of sequence

*type: integer*  
*default: 0*  
*optional: True*

**version** print version information and exit

*type: boolean*  
*default: False*  
*optional: True*



## miscellaneous

**Backend** ruff

**Author** Mark Fiers, Yogini Idnani

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** unknown

### 3.15.34 kangax

use kangax to create the suffix array lookup database for the reference genome

## Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** run kangax

## Filesets

**input\_fasta** Fasta input file

**output\_log** output log file

```
type: map
source: input_fasta
category: output
optional: {}
pattern: /*.log.txt
```

**output\_sfx** output suffix array lookup

```
type: map
source: input_fasta
category: output
optional: {}
pattern: /*.sfx
```

## Parameters

**block\_seq\_len** generated suffix blocks to hold at most this length (MB) concatenated sequences

```
type: integer
```

*default: 3300*  
*optional: True*

**color\_space** generate for colorspace (SOLiD)

*type: boolean*  
*default: False*  
*optional: True*

**extra\_params** any extra parameters

*type: string*  
*default: ""*  
*optional: True*

**help** print this help and exit

*type: boolean*  
*default: False*  
*optional: True*

**reference\_species** reference species

*type: string*  
*default: ""*  
*optional: False*

**target\_dep** generate target file only if missing or older than any independent source files

*type: boolean*  
*default: False*  
*optional: True*

**version** print version information and exit

*type: boolean*  
*default: False*

*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers, Yogini Idnani

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** unknown

### 3.15.35 map

**Execute a “map” ad-hoc analysis**

Execute one command, on a number of input files.

## Commands

**run** *no help defined*

## Filesets

**input** “map” input files

**output** “map” output files

*type: map*  
*source: input*  
*category: output*  
*optional: True*  
*pattern: /\**

## Parameters

**process** The command to execute

*type: string*  
*default: {}*  
*optional: False*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Tue Mar 29 16:34:19 2011

**Modification date** Wed Mar 30 06:02:01 2011

### 3.15.36 map2

#### Execute a “map2” ad-hoc analysis

Execute one command, on a number of input files.

#### Commands

**run** *no help defined*

#### Filesets

**input1** “map” input files set 1

**input2** “map” input files set 2

*type: map*  
*source: input1*  
*category: input*  
*optional: False*  
*pattern: \*/\**

**output** “map” output files

*type: map*  
*source: input1*  
*category: output*  
*optional: True*  
*pattern: ./\**

#### Parameters

**process** The command to execute

*type: string*  
*default: True*  
*optional: False*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Tue Mar 29 16:34:19 2011

**Modification date** Wed Mar 30 06:02:01 2011

### 3.15.37 map22

**Execute a “map22” ad-hoc analysis - two input files, two output files**

Execute one command, on a number of input files.

#### Commands

**run** *no help defined*

#### Filesets

**input1** “map” input files set 1

**input2** “map” input files set 2

*type: map*  
*source: input1*  
*category: input*  
*optional: False*  
*pattern: \*/\**

**output1** “map” output files

*type: map*  
*source: input1*  
*category: output*  
*optional: True*  
*pattern: ./\**

**output2** “map” output files

*type: map*  
*source: input1*  
*category: output*

*optional: True*  
*pattern: ./\**

## Parameters

**process** The command to execute

*type: string*  
*default: True*  
*optional: False*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Tue Mar 29 16:34:19 2011

**Modification date** Wed Mar 30 06:02:01 2011

### 3.15.38 maq\_pe

Generate alignments in SAM format given paired end reads using Maq.

## Commands

**clean** Remove all job data, not the Moa job itself

**run** run maq's fasta2bfa, fastq2bfq and map.

## Filesets

**bam\_output** bam alignment output file

*type: map*  
*source: fq\_forward\_input*  
*category: output*  
*optional: {}*  
*pattern: ./\*.bam*

**bfa\_output** BFA Index name

*type: single*

*category: other*  
*optional: {}*  
*pattern: {}*

**bfq\_forward\_output** bfq files - forward files

*type: map*  
*source: fq\_forward\_input*  
*category: output*  
*optional: {}*  
*pattern: ./\*\_1.bfq*

**bfq\_reverse\_output** bfq files - reverse files

*type: map*  
*source: fq\_forward\_input*  
*category: output*  
*optional: {}*  
*pattern: ./\*\_2.bfq*

**fa\_input** directory with reference fasta file name

**fq\_forward\_input** fastq input files directory - forward files

**fq\_reverse\_input** fastq input files directory - reverse files

*type: map*  
*source: fq\_forward\_input*  
*category: input*  
*optional: {}*  
*pattern: \*/\*\_2.fq*

**map\_output** maq map output files

*type: map*  
*source: fq\_forward\_input*  
*category: output*  
*optional: {}*  
*pattern: ./\*.map*

## Parameters

**disable\_sw** disable Smith-Waterman alignment

*type: boolean*  
*default: False*  
*optional: True*

**extra\_parameters** Any extra parameters

*type: string*  
*default: ""*  
*optional: True*

**first\_read\_len** length of the first read (<=127)s

*type: integer*  
*default: 0*  
*optional: True*

**match\_in\_colorspace** match in the colorspace

*type: boolean*  
*default: False*  
*optional: True*

**max\_dist\_read\_pairs** max distance between two paired reads s

*type: integer*  
*default: 250*  
*optional: True*

**max\_dist\_RF\_read\_pairs** max distance between two RF paired reads s

*type: integer*  
*default: 0*  
*optional: True*



**max\_mismatch\_qual\_sum** maximum allowed sum of qualities of mismatches

*type: integer*  
*default: 70*  
*optional: True*

**max\_num\_hits\_out** max number of hits to output. >512 for all 01 hits.

*type: integer*  
*default: 250*  
*optional: True*

**num\_mismatch\_24bp** number of mismatches in the first 24bp

*type: integer*  
*default: 2*  
*optional: True*

**read\_ref\_diff\_rate** rate of difference between reads and references

*type: float*  
*default: 0.001*  
*optional: True*

**sec\_read\_len** length of the second read (<=127)s

*type: integer*  
*default: 0*  
*optional: True*

**trim\_all\_reads** trim all reads (usually not recommended)

*type: boolean*  
*default: False*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers, Yogini Idnani

**Creation date** Wed Dec 03 17:06:48 2010

**Modification date** unknown

### 3.15.39 maq\_se

Generate alignments in SAM format given single end reads using Maq.

## Commands

**clean** Remove all job data, not the Moa job itself

**run** run maq's fasta2bfa, fastq2bfq and map.

## Filesets

**bam\_output** bam alignment output file

*type: map*  
*source: fq\_input*  
*category: output*  
*optional: {}*  
*pattern: /\*.bam*

**bfa\_output** BFA Index name

*type: single*  
*category: other*  
*optional: {}*  
*pattern: {}*

**bfq\_output** bfq files - forward files

*type: map*  
*source: fq\_input*  
*category: output*  
*optional: {}*  
*pattern: /\*.bfq*

**fa\_input** directory with reference fasta file name

**fq\_input** fastq input files

**map\_output** maq map output files

*type: map*  
*source: fq\_input*  
*category: output*  
*optional: {}*  
*pattern: ./\*.map*

## Parameters

**disable\_sw** disable Smith-Waterman alignment

*type: boolean*  
*default: False*  
*optional: True*

**extra\_parameters** other parameters

*type: string*  
*default: ""*  
*optional: True*

**match\_in\_colorspace** match in the colorspace

*type: boolean*  
*default: False*  
*optional: True*

**max\_mismatch\_qual\_sum** maximum allowed sum of qualities of mismatches

*type: integer*  
*default: 70*  
*optional: True*

**max\_num\_hits\_out** number of mismatches in the first 24bp

*type: integer*  
*default: 250*  
*optional: True*

**num\_mismatch\_24bp** number of mismatches in the first 24bp

*type: integer*  
*default: 2*  
*optional: True*

**read\_ref\_diff\_rate** rate of difference between reads and references

*type: float*  
*default: 0.001*  
*optional: True*

**trim\_all\_reads** trim all reads (usually not recommended)

*type: boolean*  
*default: False*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers, Yogini Idnani

**Creation date** Wed Dec 02 17:06:48 2010

**Modification date** unknown

## 3.15.40 mummer

**mummer**

Run mummer between two sequences

## Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** Run mummer

## Filesets

**input** Set 1 input fasta files

**reference** Set 1 input fasta files

## Parameters

**base** base name for all generated files

*type: {}*  
*default: out*  
*optional: True*

**breaklen** Set the distance an alignment extension will attempt to extend poor scoring regions before giving up (default 200)

*type: integer*  
*default: 200*  
*optional: True*

**genomecenter** genome center - used in the AGP file

*type: {}*  
*default: pflnz*  
*optional: True*

**gff\_source** GFF source field

*type: {}*  
*default: mumscaff*  
*optional: True*

**linker** linker sequence for the merged output sequence

*type: {}*  
*default: NNNNNNCTAGCTAGCATGNNNNNN*  
*optional: True*

**matchmode** use all matching fragments (max) or only unique matchers (mum)

*type: set*  
*default: mum*  
*optional: True*

**mum\_plot\_raw** plot an alternative visualization where mummer does not attempt to put the sequences in the correct order

*type: boolean*  
*default: False*  
*optional: True*

**organism** Organism name - used in the AGP file

*type: {}*  
*default: ""*  
*optional: True*

**taxid** Taxonomy id - used in the AGP file

*type: {}*  
*default: ""*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Wed Nov 10 07:56:48 2010

### 3.15.41 ncbi

#### Download data from NCBI

Download a set of sequences from NCBI based on a query string *ncbi\_query* and database *ncbi\_db*. This template will run only **once**, after a succesful run it creates a lock file that you need to remove to rerun

## Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** Download from NCBI

## Parameters

**db** NCBI database

*type: string*

*default: nuccore*

*optional: True*

**query** NCBI query (for example txid9397[Organism%3Aexp])

*type: string*

*default: “*

*optional: False*

**rename\_sequence** try to rename the sequence - note, this does not work if you are downloading more than one sequence

*type: boolean*

*default: False*

*optional: True*

**sequence\_name** Name of the file to write the downloaded sequences to. Use ‘from\_dir’ to have the sequence name extracted from the directory name

*type: string*

*default: out*

*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Wed Nov 10 07:56:48 2010

### 3.15.42 newjobtest

Execute a “simple” ad hoc analysis

Execute one command, No in or output files are tracked by Moa.

#### Commands

**run** *no help defined*

#### Parameters

**process** The command to execute

*type: string*  
*default: {}*  
*optional: False*

### miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Tue Mar 29 16:34:19 2011

**Modification date** Wed Mar 30 06:02:01 2011

### 3.15.43 orthomcl

**Run OrthoMCL**

Execute one command, No in or output files are tracked by Moa.

#### Commands

**run** *no help defined*

#### Parameters

**db** Db name

*type: string*  
*default: {}*  
*optional: False*



**eval** Evaluate cutoff for blast to use

*type: string*  
*default: 1e-5*  
*optional: True*

**group\_prefix** OrthoMCL prefix for group names

*type: string*  
*default: g\_*  
*optional: True*

**host** Db Host

*type: localhost*  
*default: {}*  
*optional: True*

**input\_dir** Input directory with compliant (read the manual) fasta files

*type: string*  
*default: {}*  
*optional: False*

**login** Db username

*type: string*  
*default: None*  
*optional: False*

**mcl\_i** mcl -i value

*type: float*  
*default: 1.5*  
*optional: True*

**num\_threads** Number of threads to use

*type: integer*  
*default: 4*  
*optional: True*

**pass** Db password

*type: string*  
*default: None*  
*optional: False*

**port** Db port

*type: integer*  
*default: 3306*  
*optional: True*

**prefix** OrthoMCL prefix for the database tables

*type: string*  
*default: ortho*  
*optional: True*

**vendor** Db vendor

*type: string*  
*default: mysql*  
*optional: True*

## **miscellaneous**

**Backend** ruff

**Author** Mark Fiers

**Creation date** Tue Mar 29 16:34:19 2011

**Modification date** Wed Mar 30 06:02:01 2011

### **3.15.44 project**

**Create a project**

Placeholder for a Moa Project

## Commands

**run** This template does not do anything - it is a project placeholder.

## Parameters

### miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Tue, 10 Jan 2012 14:54:39 +1300

**Modification date** Wed Nov 10 07:56:48 2010

### 3.15.45 recursive\_map

#### Recursively map a genome to the reference

Recursively map a resequence dataset against a reference genome

## Commands

**run** recursive map

## Filesets

**fq\_forward** fastq input files directory - forward

**fq\_reverse** fastq input files directory - reverse

*type: map*  
*source: fq\_forward*  
*category: input*  
*optional: True*  
*pattern: \*/\*\_2.fq*

**output** base output filename

*type: single*  
*category: output*  
*optional: True*  
*pattern: output*

## reference

*type: single*  
*category: prerequisite*  
*optional: False*  
*pattern: \*/\**

## Parameters

**iterations** no of iterations to run

*type: integer*  
*default: 3*  
*optional: True*

**param\_first** First set of parameters - get the low hanging fruit

*type: string*  
*default: -fast*  
*optional: True*

**param\_second** Second set of parameters - more sensitive

*type: string*  
*default: -very-sensitive*  
*optional: True*

**threads** Number of threads to use

*type: integer*  
*default: 4*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Fri, 08 Jun 2012 13:32:30 +1200

**Modification date** Fri, 08 Jun 2012 13:43:19 +1200

### 3.15.46 reduce

#### Execute a “reduce” ad-hoc analysis

Execute one command, on a number of input files.

#### Commands

**run** *no help defined*

#### Filesets

**input** “reduce” input files

**output** “reduce” output files

*type: single*

*category: output*

*optional: True*

*pattern: ./\**

#### Parameters

**process** The command to execute

*type: string*

*default: echo “input: {{ input|join(” ”)}}”; echo “output: {{ output }}”*

*optional: False*

#### miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Tue Mar 29 16:34:19 2011

**Modification date** Wed Mar 30 06:02:01 2011

### 3.15.47 samtools\_pileup

Print the alignment in the pileup format.

## Commands

**clean** Remove all job data, not the Moa job itself

**run** run samtools pileup command

## Filesets

**fasta** reference fasta file

*type: single*  
*category: prerequisite*  
*optional: True*  
*pattern: \*/\*.fasta*

**input** bam or sam files

**output**

*type: map*  
*source: input*  
*category: output*  
*optional: {}*  
*pattern: ./\*.pileup*

**output\_bam**

*type: map*  
*source: input*  
*category: output*  
*optional: {}*  
*pattern: ./\*.sorted*

## Parameters

**cap\_mapQ\_at** cap mapping quality at INT

*type: integer*  
*default: 60*  
*optional: True*

**extra\_params** any extra parameters

*type: string*  
*default: “*

*optional: True*

**filter\_read\_bits** filtering reads with bits in INT

*type: integer*

*default: 1796*

*optional: True*

**input\_is\_SAM** the input is in SAM

*type: boolean*

*default: False*

*optional: True*

**num\_haplotypes** number of haplotypes in the sample (for -c/-g)

*type: integer*

*default: 2*

*optional: True*

**out\_2nd\_best** output the 2nd best call and quality

*type: boolean*

*default: False*

*optional: True*

**out\_GLFv3\_format** output in the GLFv3 format (suppressing -c/-i/-s)

*type: boolean*

*default: False*

*optional: True*

**out\_maq\_consensus** output the maq consensus sequence

*type: boolean*

*default: False*

*optional: True*

**phred\_prob\_indel** phred prob. of an indel in sequencing/prep. (for -c/-g)

*type: integer*  
*default: 40*  
*optional: True*

**print\_variants\_only** print variants only (for -c)

*type: boolean*  
*default: False*  
*optional: True*

**prior\_diff\_haplotypes** phred prob. of an indel in sequencing/prep. (for -c/-g)

*type: float*  
*default: 0.001*  
*optional: True*

**prior\_indel\_haplotypes** number of haplotypes in the sample (for -c/-g)

*type: float*  
*default: 0.00015*  
*optional: True*

**show\_lines\_indels** only show lines/consensus with indels

*type: boolean*  
*default: False*  
*optional: True*

**simple\_pileup\_format** simple (yet incomplete) pileup format

*type: boolean*  
*default: False*  
*optional: True*

**theta\_maq\_model** number of haplotypes in the sample (for -c/-g)



*type: float*  
*default: 0.85*  
*optional: True*

**use\_SOAPsnp\_model** use the SOAPsnp model for SNP calling

*type: boolean*  
*default: False*  
*optional: True*

## **miscellaneous**

**Backend** ruff

**Author** Yogini Idnani, Mark Fiers

**Creation date** Wed Dec 15 17:06:48 2010

**Modification date** unknown

### **3.15.48 simple**

**Execute a “simple” ad hoc analysis**

Execute one command, No in or output files are tracked by Moa.

## **Commands**

**run** *no help defined*

## **Parameters**

**process** The command to execute

*type: string*  
*default: {}*  
*optional: False*

## **miscellaneous**

**Backend** ruff

**Author** Mark Fiers

**Creation date** Tue Mar 29 16:34:19 2011

**Modification date** Wed Mar 30 06:02:01 2011

### 3.15.49 smalt\_pe

Run SMALT on an set of input files (query) vs a database index.

#### Commands

**clean** Remove all job data, not the Moa job itself

**run** Execute SMALT with with paired-end fastq

#### Filesets

**db** The (basename of the) smalt database to use.

*type: single*  
*category: prerequisite*  
*optional: False*  
*pattern: ../10.smaltdb/db*

**fasta** reference fasta file

*type: single*  
*category: prerequisite*  
*optional: False*  
*pattern: \*.fasta*

**fq\_forward\_input** fastq input files directory - forward

**fq\_reverse\_input** fastq input files directory - reverse

*type: map*  
*source: fq\_forward\_input*  
*category: input*  
*optional: True*  
*pattern: \*/\*\_2.fq*

**output** output BAM file (automatically converted & filtered for reads that to not map)

*type: map*  
*source: fq\_forward\_input*  
*category: output*  
*optional: {}*

*pattern: ./\*.sam*

## Parameters

**extra\_params** extra parameters to feed to smalt

*type: string*  
*default: ""*  
*optional: True*

**max\_insertsize** Maximum allowed insertsize

*type: integer*  
*default: 250*  
*optional: True*

**min\_insertsize** Minimum allowed insertsize

*type: integer*  
*default: 1*  
*optional: True*

**output\_format** output format (sam or samsoft)

*type: {}*  
*default: sam*  
*optional: True*

**pairtype** pair type (pe: fr/illumina short; mp: rf/illumina mate pairs or pp: ff

*type: {}*  
*default: pe*  
*optional: True*

**threads** No threads to use

*type: int*  
*default: 4*

*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Tue, 27 Mar 2012 10:05:40 +1300

**Modification date** Tue, 27 Mar 2012 10:31:09 +1300

## 3.15.50 smaltdb

### Smalt index builder

Builds a smalt index from a reference sequence

## Commands

**clean** Remove all job data, not the Moa job itself, note that this must be implemented by the template.

**run** Create the smalt index

## Filesets

**input** Input fasta file for the smalt database

*type: single*  
*category: input*  
*optional: False*  
*pattern: \*/\*.fasta*

**output** database name to create

*type: single*  
*category: output*  
*optional: {}*  
*pattern: db*

## Parameters

**word\_length** word length

*type: int*

*default: 10*  
*optional: True*

**word\_spacing** word spacing

*type: int*  
*default: 6*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Wed Nov 10 07:56:48 2010

**Modification date** Wed Dec 09 07:56:48 2010

### 3.15.51 soapdenovo\_pe

Run Soapdenovo

## Commands

**clean** Remove all job data

**run** Execute soapdenovo in paired-end mode

## Filesets

**fq\_forward** fastq input files directory - forward

**fq\_reverse** fastq input files directory - reverse

*type: map*  
*source: fq\_forward*  
*category: input*  
*optional: True*  
*pattern: \*/\*\_2.fq*

**output** soap denovo output file

*type: single*  
*category: output*

*optional: True*  
*pattern: output.scafSeq*

## Parameters

**avg\_insert** library insert size

*type: integer*  
*default: 200*  
*optional: {}*

**executable\_not\_used\_anymore** which executable to use (SOAPdenovo-127mer, SOAPdenovo-31mer or SOAPdenovo-63mer)

*type: {}*  
*default: SOAPdenovo-31mer*  
*optional: True*

**kmer** kmer size

*type: integer*  
*default: 31*  
*optional: True*

**skip\_config\_file** skip automatic config file generation - if you skip this, make sure that you have a soap.config configuration file in the current directory

*type: boolean*  
*default: False*  
*optional: True*

**threads** no threads to use

*type: integer*  
*default: 8*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Mon, 21 Nov 2011 12:47:16

**Modification date** Mon, 21 Nov 2011 12:47:22

### 3.15.52 soapdenovo\_postprocess

Run Soapdenovo

#### Commands

**run** Postprocess - run GapCloser & SSpace

#### Filesets

**fq\_forward** fastq input files directory - forward

**fq\_reverse** fastq input files directory - reverse

*type: map*  
*source: fq\_forward*  
*category: input*  
*optional: True*  
*pattern: \*/\*\_2.fq*

**input** input scaffold to process

*type: single*  
*category: input*  
*optional: False*  
*pattern: {}*

**output** output file to generate

*type: single*  
*category: output*  
*optional: True*  
*pattern: final.fasta*

## Parameters

**avg\_insert** library insert size

*type: integer*  
*default: 200*  
*optional: {}*

**noruns** no times to run gapcloser & SSPace

*type: integer*  
*default: 2*  
*optional: True*

**run\_ssplace** run SSPace? use

*type: boolean*  
*default: True*  
*optional: True*

**ssplace\_executable** SSPace executable

*type: {}*  
*default: SSPACE\_Basic\_v2.0.pl*  
*optional: True*

**ssplace\_extra\_variables** Extra variables to pass to Sspace

*type: {}*  
*default: ""*  
*optional: True*

**threads** no threads to use

*type: integer*  
*default: 8*  
*optional: True*



## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Mon, 21 Nov 2011 12:47:16

**Modification date** Mon, 21 Nov 2011 12:47:22

### 3.15.53 statsidx

Retrieve and print stats from BAM file to an index file

## Commands

**clean** Remove all job data, not the Moa job itself

**run** run samtools idxstats

## Filesets

**input** bam input files directory - forward files

**output**

*type: map*

*source: input*

*category: output*

*optional: {}*

*pattern: ./\*.index*

## Parameters

### miscellaneous

**Backend** ruff

**Author** Yogini Idnani, Mark Fiers

**Creation date** Wed Dec 08 17:06:48 2010

**Modification date** unknown

### 3.15.54 sync

**Sync directories**

Create this directory in sync with another directory

## Commands

**run** Sync!

## Parameters

**ignore** ignore these names (space separated list)

*type: {}*  
*default: ""*  
*optional: True*

**original** The local directory to use as a source. If the target (based on what is in the source) does not exist, this directory is copied. If the target exists - only the configuration is copied, and all directory contents are left alone. If this parameter is omitted, the directory with the most recently changed moa configuration.

*type: string*  
*default: {}*  
*optional: True*

**recursive** copy the jobs/config recursively

*type: boolean*  
*default: False*  
*optional: True*

**source** The directory to keep in sync with. If not specified, this template just keeps local directory synced

*type: string*  
*default: {}*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Thu, 30 Jun 2011 21:26:19

**Modification date** Thu, 30 Jun 2011 21:25:53

### 3.15.55 unittest

Template used in testing - has no other purpose

#### Commands

**clean** Remove all job data

**prepare** prepare for the unittest

**run** Prepare & Run

**run** delegates execution to: **prepare**, **run2**

**run2** actually run

#### Filesets

**input\_1** Input file set 1

**input\_2** Input file set 2

*type: map*  
*source: input\_1*  
*category: input*  
*optional: {}*  
*pattern: in2/\*\_2.txt*

**output** output files

*type: map*  
*source: input\_1*  
*category: output*  
*optional: {}*  
*pattern: ./\*.out*

#### Parameters

**test\_string** Test string values

*type: string*  
*default: {}*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Yogini Idnani, Mark Fiers

**Creation date** Wed Nov 25 17:06:48 2010

**Modification date** unknown

### 3.15.56 wget

#### wget

Use WGET to download files. This template has two modi, one is set wget\_mode to mirror data, in which case both wget\_url and wget\_pattern (default \*) are used. The other modus is wget\_mode=get, when one file defined by wget\_url is downloaded. In the mirror mode it is possible to download only those files that are newer as the files already downloaded by using the wget\_timestamp parameter

#### Commands

**run** Download

#### Parameters

**pass** Password for the remote site (note - this is not very safe, the password will be stored in plan text

*type: password*  
*default: ""*  
*optional: True*

**url** The url of the file to download

*type: string*  
*default: {}*  
*optional: False*

**user** Username for the remote site

*type: string*  
*default: ""*  
*optional: True*

## miscellaneous

**Backend** ruff

**Author** Mark Fiers

**Creation date** Thu, 02 Jun 2011 10:22:31 +1200

**Modification date** Thu, 02 Jun 2011 10:22:53 +1200

## 3.16 Moa API

### 3.16.1 moa.actor

‘Simple’ wrapper around subprocess to execute code

`moa.actor.async(f)`  
decorator designating an actor to be asynchronous

`moa.actor.getLastStderr(job)`  
Get the last stderr

`moa.actor.getLastStdout(job)`  
Get the last stdout

`moa.actor.getRecentOutDir(job)`  
Return the most recent output directory

`moa.actor.simpleRunner(wd, cl, conf={}, **kwargs)`  
Don’t think - just run - here & now

what does this function do? - put env in the environment - Execute the commandline (in cl) - store stdout & stderr in log files - return the rc

`moa.actor.sync(f)`  
decorator designating an actor to be synchronous

### 3.16.2 moa.commands

Handle Moa commands (i.e. anything that you can run as *moa COMMAND* on the commandline)

**exception** `moa.exceptions.CannotGetAFileLock(f)`  
Cannot get a file lock

**exception** `moa.exceptions.MoaCommandDoesNotExist`  
Command does not exists?

**exception** `moa.exceptions.MoaDirNotWritable`  
Moa directory is not writable

**exception** `moa.exceptions.MoaFileError`  
Error handling a file

**exception** `moa.exceptions.MoaInvalidCommandLine`  
Invalid command line

**exception** `moa.exceptions.MoaPermissionDenied`

Permission denied - you do not have the rights to perform this operation

**exception** `moa.exceptions.NotAMoaDirectory` (*dir*)

This is not a moa directory

**exception** `moa.exceptions.NotAMoaTemplate` (*template*)

This is not a valid moa template

### 3.16.3 fileset - define sets of in&output files

`moa.filesets.render` (*job*)

render all filesets - i.e. figure out what files belong in what sets

### 3.16.4 moa.job

**class** `moa.job.Job` (*wd*)

Class defining a single job

Note - in the moa system, there can be only one current job - many operations try to access the job in sysConf

```
>>> wd = tempfile.mkdtemp()
>>> job = Job(wd)
>>> assert(isinstance(job, Job))
>>> assert(job.template.name == 'nojob')
```

**checkCommands** (*command*)

Check command, and rearrange if there are delegates.

```
>>> job = newTestJob('unittest')
```

```
## >>> assert(job.template.commands.run.delegate == ['prepare', 'run2']) ## >>> as-
assert(job.checkCommands('run2') == ['run2']) ## >>> assert(job.checkCommands('run') ==
['prepare', 'run2']) ## >>> assert(job.checkCommands('prepare') == ['prepare'])
```

**Parameters** **commands** (*list of strings*) – The list of commands to check

**Returns** The checked list of commands

**Return type** list of strings

**checkConfDir** ()

Check if the configuration directory exists. If not create it.

```
>>> job = newTestJob('unittest')
>>> confdir = os.path.join(job.wd, '.moa')
>>> assert(os.path.exists(confdir))
>>> import shutil
>>> shutil.rmtree(confdir)
>>> assert(os.path.exists(confdir) == False)
>>> job.checkConfDir()
>>> assert(os.path.exists(confdir))
```

**defineCommands** (*commandparser*)

Register template commands with the argparser

**defineOptions** (*parser*)

Set command line options - deferred to the backend - PER COMMAND

```
>>> job = newTestJob('unittest')
>>> import optparse
>>> parser = optparse.OptionParser()
>>> job.defineOptions(parser)
```

**execute** (*job, args, \*\*kwargs*)

Execute *command* in the context of this job. Execution is always deferred to the backend

#Note: this is the function that will be called from argparse #Note: Uncertain how to test verbose & silent

**Parameters**

- **verbose** (*Boolean*) – output lots of data
- **silent** (*Boolean*) – output nothing

**finishExecute** ()

Finish the run!

**getFiles** ()

Return all moa files - i.e. all files crucial to this job.

**hasCommand** (*command*)

Check if this job defines a certain command

**Warning:** THIS METHOD DOES NOT WORK PROPERLY YET

```
>>> job = newTestJob('unittest')
>>> assert (job.hasCommand('run'))
```

```
### >>> assert(job.hasCommand('dummy'))
```

**init2** ()

Continue initialization

**initialize** ()

Initialize a new job in the current wd

**isMoa** ()

Check if this is a Moa directory - Currently, this needs to be overridden

TODO: check if this ever gets called

**loadBackend** ()

load the backend

**loadTemplate** ()

Load the template for this job, based on what configuration can be found

**prepareExecute** ()

Give this job a chance to prepare for execution.

**refreshTemplate** ()

Reload the template into the local .moa/template.d directory

```
>>> job = newTestJob('unittest')
>>> templ = os.path.join(job.confDir, 'template.d', 'unittest.jinja2')
>>> assert(os.path.exists(templ))
>>> os.unlink(templ)
>>> assert(not os.path.exists(templ))
>>> job.refreshTemplate()
>>> assert(os.path.exists(templ))
```

**run\_hook** (*hook*, *\*\*kwargs*)  
Shortcut to run a job plugin hook

**setTemplate** (*name*, *provider=None*)  
Set a new template for this job

```
>>> job = newTestJob('unittest')
>>> job.setTemplate('simple')
>>> afile = os.path.join(job.confDir, 'template.d', 'simple.jinja2')
>>> assert(os.path.exists(afile))
```

`moa.job.newJob` (*job*, *template*, *title*, *parameters=[]*, *provider=None*)

Create a new job in the wd and return the proper job object currently only makefile jobs are supported - later we'll scan the template, and instantiate the proper job type

#### Parameters

- **job** – Job object to fill - needs only wd set.
- **template** (*String*) – Template name for this job
- **parameters** (*list of (key, value) tuples*) – A list of parameters to set for this job

**Return type** instance of `moa.job.Job`

`moa.job.newTestJob` (*template*, *title='Test job'*, *provider=None*)

for testing purposes - creates a temporary directory and uses that to instantiate a job. This function returns the job object created

```
>>> job = newTestJob(template = 'simple', title='test title')
>>> assert(isinstance(job, Job))
>>> assert(os.path.exists(job.wd))
>>> assert(job.conf.title == 'test title')
>>> assert(os.path.exists(os.path.join(job.wd, '.moa')))
>>> assert(os.path.exists(os.path.join(job.wd, '.moa', 'template')))
```

```
### >>> assert(job.template.name == 'simple')
```

**Returns** the created job

**Return type** instance of `moa.job.Job`

### 3.16.5 moa.jobConf

moa job configuration

**class** `moa.jobConf.JobConf` (*job*)

to distinguish between attributes of this object & proper job configuration parameters



**doNotCheck = None**

these fields are not be type-checked

**doNotSave = None**

these fields are not to be saved

**getRendered** (*key*)

Get the rendered value of this key

**isEmpty** ()

Check if the config is empty is empty

**isPrivate** (*k*)

Is this a private variable? can be locally defined or in the template definition

**keys** ()

return a dict with all known parameters and values, either defined in the job configuration of the template

**load** (*confFile*, *delta=None*)

Load a configuration file

**Parameters** **delta** – if a value appears to be a relative path, try to correct for this. Currently this only works for files that exist. i.e.

**private = None**

these fields are private (i.e. not to be displayed by default)

**save** ()

Save the conf to disk

**setRecursiveVar** (*k*, *v*)

Register a recursive variable

**class** `moa.logger.MoaFormatter` (*fmt=None*, *datefmt=None*)

A somewhat more advanced formatter

**format** (*record*)

Defines two extra fields in the record class, upon formatting: - **visual**, a visual indication of the severity of the message - **tb**, a formatted traceback, used when sending mail @param **record**: the log message record

### 3.16.6 moa.sysConf

Store Moa wide configuration

### 3.16.7 moa.ui

communicate information to the user

`moa.ui.askUser` (*parameter*, *default=''*, *extra\_history=None*)

**Parameters**

- **parameter** – paramter to ask value of
- **default** – default value - if absent use the last history item

- **xtra\_history** – extra history file to show to the user

### 3.16.8 moa.utils

A set of random utilities used by Moa

`moa.utils.deprecated(func)`

Decorator function to flag a function as deprecated

**Parameters** `func` – any function

`moa.utils.flog(f)`

A simple logger - uses the `moa.logger` code to log the calling function. Use as a decorator:

```
@moa.utils.flog
def any_function(*args):
    ...
```

This is for debugging purposes (obviously)

**Parameters** `func` – Any python function

`moa.utils.getCwd()`

Do not use `os.getcwd()` - need to make sure symbolic links do not get dereferenced

hijacked some code from: <http://stackoverflow.com/questions/123958/how-to-get-set-logical-directory-path-in-python>

`moa.utils.getMoaBase()`

Return MOABASE - the directory where Moa is installed. This function also sets an environment variable `MOABASE`

```
>>> d = getMoaBase()
>>> assert os.path.isdir(d)
>>> assert os.path.isfile(os.path.join(d, 'README'))
>>> assert os.path.isdir(os.path.join(d, 'lib'))
```

**Return type** string (path)

`moa.utils.getProcessInfo(pid)`

Return some info on a process

`moa.utils.moaDirOrExit(job)`

Check if the job contains a proper Moa job, if not, exit with an error message and a non-zero exit code.

**Parameters** `job` – An instance of `moa.job.Job`

`moa.utils.niceRunTime(d)`

Nice representation of the run time d is time duration string

`moa.utils.printstack(func)`

Decorator function to print stack

**Parameters** `func` – any function

`moa.utils.removeIndent(txt)`

Removes indentation from a txt - for use by `moa.args` and `moa.api`

`moa.utils.sendmail` (*server, sender, recipient, subject, message*)  
Send an email.

`moa.utils.simple_decorator` (*decorator*)

This decorator can be used to turn simple functions into well-behaved decorators, so long as the decorators are fairly simple. If a decorator expects a function and returns a function (no descriptors), and if it doesn't modify function attributes or docstring, then it is eligible to use this. Simply apply `@simple_decorator` to your decorator and it will automatically preserve the docstring and function attributes of functions to which it is applied.

Note; I got this code from somewhere, but forgot where exactly. This seems the most likely source:

<http://svn.navi.cx/misc/trunk/djblets/djblets/util/decorators.py>

### 3.16.9 moa.template

#### moa.template

Store information on a template. This module is also responsible for retrieving template information.

`moa.template.initTemplate` (*\*args, \*\*kwargs*)

`moa.template.installTemplate` (*wd, tName, provider=None*)

Initialize the template - this means - try to figure out where the template came from & copy the template files into *job/.moa/template* & *job/.moa/template.d/extra*.

Currently all templates come from the moa repository. In the future, multiple sources must be possible

```
>>> import tempfile
>>> wd = tempfile.mkdtemp()
>>> installTemplate(wd, 'simple')
>>> templateFile = os.path.join(wd, '.moa', 'template')
>>> adhocFile = os.path.join(wd, '.moa', 'template.d', 'simple.jinja2')
>>> assert os.path.exists(templateFile)
>>> assert os.path.exists(adhocFile)
```

`moa.template.refresh` (*wd*)

Refresh the template - try to find out what the template is from `{{wd}}/.moa/template.meta`. If that doesn't work, revert to the default template. If default is not specified - exit with an error

```
>>> import tempfile
>>> wd = tempfile.mkdtemp()
>>> installTemplate(wd, 'simple')
>>> templateFile = os.path.join(wd, '.moa', 'template')
>>> adhocFile = os.path.join(wd, '.moa', 'template.d', 'simple.jinja2')
>>> os.unlink(adhocFile)
>>> os.unlink(templateFile)
>>> assert(not os.path.exists(templateFile))
>>> assert(not os.path.exists(adhocFile))
>>> refresh(wd)
>>> assert(os.path.exists(templateFile))
>>> assert(os.path.exists(adhocFile))
```

## moa.template.template

Store information on a template. This module is also responsible for retrieving template information.

**class** `moa.template.template.Template(wd)`

Template extends Yaco

**getRaw()**

Return a Yaco representation of the yaml-template, without any of this Template processing. This is really useful when processing a template that needs to be written back to disk

```
>>> import moa.job
>>> job = moa.job.newTestJob(template='simple')
>>> raw = job.template.getRaw()
>>> assert(isinstance(raw, Yaco.Yaco))
>>> assert(raw.has_key('parameters'))
```

**loadMeta()**

Load the template meta data for this job, based on what configuration can be found

### 3.16.10 moa.template.provider

#### moa.provider.core

Provides templates from the Moa package.

### 3.16.11 moa.backend

#### Ruff

Ruffus (and Jinja) Backend

**members**

### 3.16.12 moa.plugin

### 3.16.13 Yaco

Yaco provides a *dict* like structure that can be serialized to & from `yaml`. Yaco objects behave as dictionaries but also allow attribute access (loosely based on this [recipe <http://code.activestate.com/recipes/473786/>](http://code.activestate.com/recipes/473786/)). Sublevel dictionaries are automatically converted to Yaco objects, allowing sublevel attribute access, for example:

```
>>> x = Yaco()
>>> x.test = 1
>>> x.sub.test = 2
>>> x.sub.test
2
```

Note that sub-dictionaries do not need to be initialized. This has as a consequence that requesting uninitialized items automatically return an empty Yaco object (inherited from a dictionary).

Yaco can be found in the [Python package index](#) and is also part of the [Moa source distribution](#)

## Autogenerating keys

An important feature (or annoyance) of Yaco is the auto generation of keys that are not present (yet). For example:

```
>>> x = Yaco()
>>> x.a.b.c.d = 1
>>> assert(x.a.b.c.d == 1)
```

works - *a*, *b* and *c* are assumed to be *Yaco* dictionaries and *d* is give value *1*. This makes populating data structures easy.

It might also generate some confusion when querying for keys in the Yaco structure - if a key does not exists, it automatically comes back as an empty *dict* or *Yaco* object (renders as *{}*). This means that if it is easy to check if a certain 'branch' of a Yaco datastructure exists:

```
>>> x = Yaco()
>>> assert(not x.a.b)
```

but now the following works as well:

```
>>> assert('a' in x)
>>> assert('b' in x.a)
```

So, a safe way to test a data structure, without introducing extra branches is:

```
>>> x = Yaco()
>>> assert(not 'a' in x)
```

Todo: Need to find a more elegant way of testing without introducing data structures

**class** `Yaco.PolyYaco` (*name='PY', files=[], pattern='\*.config', leaf=''*)

A meta object that allows a composite Yaco object to be loaded from any number of different files which are kept as a stack of Yaco objects. If looking for a value, this object will check each of the layers in the stack and return the first value that it comes across.

Changes are only made to the toplevel object.

The goal is to have multiple configuration files, for example in:

```
/location/to/python/package/etc/config.yaml
/etc/APPLICATION.yaml
~/.config/APPLICATION/config.yaml
```

and have values in the latter file override those in the former. Saving changed values will also be done to the latter, but system and application wide settings can be maintained as well (manually for the time being).

**load** (*leaf, files, pattern*)

**class** `Yaco.Yaco` (*data={}, leaf=None*)

**Originated from:** <http://code.activestate.com/recipes/473786/>

```
>>> v= Yaco()
>>> v.a = 1
>>> assert(v.a == 1)
>>> assert(v['a'] == 1)
>>> v= Yaco({'a':1})
```

```
>>> assert(v.a == 1)
>>> assert(v['a'] == 1)
```

**get\_data()**

Prepare & parse data for export

```
>>> y = Yaco()
>>> y.a = 1
>>> y.b = 2
>>> y._c = 3
>>> assert(y._c == 3)
>>> d = y.get_data()
>>> assert('a' in d)
>>> assert('b' in d)
>>> assert(not 'c' in d)
>>> y._private = ['b']
>>> d = y.get_data()
>>> assert('a' in d)
>>> assert(not 'b' in d)
>>> assert(not '_c' in d)
```

**load(from\_file, leaf=None)**

Load this dict from\_file

Note - it can load the file into a leaf, instead of the root of this Yaco structure. Note - the leaf variable is a string, but may contain dots (which are automatically interpreted)

```
>>> import tempfile
>>> tf = tempfile.NamedTemporaryFile(delete=True)
>>> tf.close()
>>> x = Yaco({'a': [1,2,3, [1,2,3, {'d' : 4}]],
...         'b': 4, 'c': '5', 'uni' : "Aπ"})
>>> x.save(tf.name)
>>> y = Yaco()
>>> y.load(tf.name)
>>> assert(y.a[3][3].d == 4)
>>> assert(sys.version_info[0] == 2 or y.uni == "Aπ")
```

**pretty()**

Return data as a pprint.pformatted string

**save(to\_file, doNotSave=[])****simple()**

return a simplified representation of this Yaco struct - remove Yaco from the equation - and all object reference. Leave only bool, float, str, lists, tuples and dicts

```
>>> x = Yaco()
>>> x.y.z = 1
>>> assert(isinstance(x.y, Yaco))
>>> s = x.simple()
>>> assert(s['y']['z'] == 1)
>>> assert(isinstance(s['y'], dict))
>>> assert(not isinstance(s['y'], Yaco))
```

**soft\_update(data)**

As update - but only update keys that do not have a value.

Note - lists are completely

```
>>> d1 = {'a' : [1,2,3,{ 'b': 12}], 'd' : { 'e': 72}}
>>> d2 = {'a' : [2,3,4,{ 'b': 12}], 'd' : { 'e': 73, 'f': 18}, 'c' : 18}
>>> v = Yaco(d1)
>>> assert(v.a[2] == 3)
>>> assert(v.d.e == 72)
>>> v.soft_update(d2)
>>> assert(v.d.e == 72)
>>> assert(v.d.f == 18)
>>> assert(v.a[2] == 3)
```

**update** (*data*)

```
>>> v = Yaco({'a' : [1,2,3,{ 'b' : 12}]})
>>> assert(v.a[3].b == 12)
```

```
>>> v = Yaco({'a' : [1,2,3,[1,{ 'b' : 12}]]})
>>> assert(v.a[3][1].b == 12)
```

**class** `Yaco.YacoDir` (*dirname*, *pattern*=`'*.config'`)

As Yaco, but load all files in a directory on top of each other.

Order of loading is the alphanumerical sort of filenames

files in subdirectories are loaded into leaves e.g. a file in `/tmp/test/sub/a.yaml` with only (`x=1`) will end up as follows:

```
y = YacoDir('/tmp/test') y.sub.x == 1
```

Note, YacoDir will try to cache itself in a `.yacodir.cache` file in the root of the *dirname* if the modification date of this file is the same as the directory - that will be loaded instead.

**load** (*dirname*, *pattern*)

Load from the defined directory

**save** ()

Save is disabled.

**class** `Yaco.YacoFile` (*filename*)

As Yaco, but loads from a file - or returns an empty object if it cannot find the file

**load** ()

Load from the defined filename

**save** ()

Load from the defined filename

### 3.16.14 fist

#### Filesets

Handle & manipulate sets of files

This module aims at providing classes to handle and manipulate sets of files. Two simple examples are a simple set containing one file (*fist.fistSingle*) or a *glob* based set of files (*fist.fistFileset*). A more complicated example is *fistMapset* that maps another fileset based on a pattern.

Each fileset inherits from *list* - hence fist filesets behave as lists.

Future work should allow the definition of remote filesets (for example over http or ssh).

Each fist class is instantiated with a url defining the file(set). In the case of `fist.fistFileset` this url contains a globbing characters:

```
fs = fist.fistFileset('/tmp/*.txt')
```

This fileset object contains a list with all `*.txt` files in `/tmp`. Subsequently it is possible to map this set

**class** `fist.fistCore` (*url*, *context=None*)

Core class for all fist classes

**resolve** ()

This function needs to be overridden context

**class** `fist.fistFileset` (*url*, *context=None*)

Most basic set of files - handle a set of files described by a single URI with wildcards, for example:

```
* `*.txt`  
* `../*.txt`  
* `file:///home/name/data/*.txt`
```

```
>>> f = fistFileset('*.txt')  
>>> assert(f.path=='.')  
>>> assert(f.glob=='*.txt')  
>>> assert(f.path=='.')  
>>> assert(f.glob=='*.txt')  
>>> f = fistFileset('/tmp')  
>>> assert(f.path=='/tmp')  
>>> assert(f.glob=='*')  
>>> f = fistFileset('/tmp/*.txt')  
>>> assert(f.path=='/tmp')  
>>> assert(f.glob=='*.txt')  
>>> f = fistFileset('../*.txt')  
>>> assert(f.path=='..')  
>>> assert(f.glob=='*.txt')  
>>> f = fistFileset(os.path.join(wd, 'in', '*.txt'))  
>>> f.resolve()  
>>> assert(len(f) == 100)  
>>> f = fistFileset(os.path.join(wd, 'in', 'in1*.txt'))  
>>> f.resolve()  
>>> assert(len(f) == 10)  
>>> f = fistFileset('~/*')  
>>> f.resolve()  
>>> assert(len(f) > 0)
```

**class** `fist.fistMapset` (*url*, *context=None*)

Map set - map a fileset based on a target uri

```
>>> f = fistFileset(os.path.join(wd, 'in', '*'))  
>>> f.resolve()  
>>> assert(len(f) == 100)  
>>> ##  
>>> ## Null mapping  
>>> ##  
>>> m = fistMapset('*/*')  
>>> m.resolve(f)  
>>> assert(len(m) == 100)  
>>> assert(os.path.join(wd, 'in/in18.txt') in m)
```



```

>>> ##
>>> ## simple folder mapping
>>> ##
>>> m = fistMapset('out/*')
>>> m.resolve(f)
>>> assert(len(m) == 100)
>>> assert('out/in18.txt' in m)
>>> ##
>>> ## simple folder mapping
>>> ##
>>> m = fistMapset('./*')
>>> m.resolve(f)
>>> assert(len(m) == 100)
>>> assert('./in18.txt' in m)
>>> ##
>>> ## simple folder & mapping & extension append
>>> ##
>>> m = fistMapset('out/*.out')
>>> m.resolve(f)
>>> assert(len(m) == 100)
>>> assert('out/in18.txt.out' in m)
>>> ##
>>> ## New from fileset - now with a pattern defining the extension
>>> ##
>>> f = fistFileset(os.path.join(wd, 'in', '*.txt'))
>>> f.resolve()
>>> ##
>>> ## extension mapping
>>> ##
>>> m = fistMapset('out/*.out')
>>> m.resolve(f)
>>> assert(len(m) == 100)
>>> assert('out/in18.out' in m)
>>> ##
>>> ## New from fileset - now with a pattern defining file glob &
>>> ## extension
>>> ##
>>> f = fistFileset(os.path.join(wd, 'in', 'in*.txt'))
>>> f.resolve()
>>> ##
>>> ## more complex filename mapping
>>> ##
>>> m = fistMapset('out/test*.out')
>>> m.resolve(f)
>>> assert(len(m) == 100)
>>> assert('out/test18.out' in m)
>>> ##
>>> ## mapping keeping the extension the same
>>> ##
>>> m = fistMapset('out/test*.txt')
>>> m.resolve(f)
>>> assert(len(m) == 100)
>>> assert('out/test18.txt' in m)

```

**resolve** (*mapFrom*)

Resolve the mapped set based on a input fileSet

**resolver** (*mapFrom*, *list*)

map all files in the incoming list

**class** `fist.fistSingle` (*url*, *context=None*)

Represents a single file

**init** ()

Assuming the url is a single file

---

## Indices and tables

---

- `genindex`
- `modindex`
- `search`



## f

`fist`, [139](#)

## m

`moa.actor`, [129](#)

`moa.backend.ruff`, [136](#)

`moa.cli`, [129](#)

`moa.commands`, [129](#)

`moa.exceptions`, [129](#)

`moa.filesets`, [130](#)

`moa.job`, [130](#)

`moa.jobConf`, [132](#)

`moa.logger`, [133](#)

`moa.sysConf`, [133](#)

`moa.template`, [135](#)

`moa.template.provider.core`, [136](#)

`moa.template.template`, [135](#)

`moa.ui`, [133](#)

`moa.utils`, [134](#)

`moa.version`, [135](#)

## y

`Yaco`, [136](#)



**A**

askUser() (in module moa.ui), 133  
async() (in module moa.actor), 129

**C**

CannotGetAFileLock, 129  
checkCommands() (moa.job.Job method), 130  
checkConfDir() (moa.job.Job method), 130

**D**

defineCommands() (moa.job.Job method), 130  
defineOptions() (moa.job.Job method), 131  
deprecated() (in module moa.utils), 134  
doNotCheck (moa.jobConf.JobConf attribute), 132  
doNotSave (moa.jobConf.JobConf attribute), 133

**E**

execute() (moa.job.Job method), 131

**F**

finishExecute() (moa.job.Job method), 131  
fist (module), 139  
fistCore (class in fist), 140  
fistFileset (class in fist), 140  
fistMapset (class in fist), 140  
fistSingle (class in fist), 142  
flog() (in module moa.utils), 134  
format() (moa.logger.MoaFormatter method), 133

**G**

get\_data() (Yaco.Yaco method), 138  
getCwd() (in module moa.utils), 134  
getFiles() (moa.job.Job method), 131  
getLastStderr() (in module moa.actor), 129  
getLastStdout() (in module moa.actor), 129  
getMoaBase() (in module moa.utils), 134  
getProcessInfo() (in module moa.utils), 134  
getRaw() (moa.template.template.Template method), 136

getRecentOutDir() (in module moa.actor), 129  
getRendered() (moa.jobConf.JobConf method), 133

**H**

hasCommand() (moa.job.Job method), 131

**I**

init() (fist.fistSingle method), 142  
init2() (moa.job.Job method), 131  
initialize() (moa.job.Job method), 131  
initTemplate() (in module moa.template), 135  
installTemplate() (in module moa.template), 135  
isEmpty() (moa.jobConf.JobConf method), 133  
isMoa() (moa.job.Job method), 131  
isPrivate() (moa.jobConf.JobConf method), 133

**J**

Job (class in moa.job), 130  
JobConf (class in moa.jobConf), 132

**K**

keys() (moa.jobConf.JobConf method), 133

**L**

load() (moa.jobConf.JobConf method), 133  
load() (Yaco.PolyYaco method), 137  
load() (Yaco.Yaco method), 138  
load() (Yaco.YacoDir method), 139  
load() (Yaco.YacoFile method), 139  
loadBackend() (moa.job.Job method), 131  
loadMeta() (moa.template.template.Template method), 136  
loadTemplate() (moa.job.Job method), 131

**M**

moa.actor (module), 129  
moa.backend.ruff (module), 136  
moa.cli (module), 129  
moa.commands (module), 129

- [moa.exceptions \(module\), 129](#)
- [moa.filesets \(module\), 130](#)
- [moa.job \(module\), 130](#)
- [moa.jobConf \(module\), 132](#)
- [moa.logger \(module\), 133](#)
- [moa.sysConf \(module\), 133](#)
- [moa.template \(module\), 135](#)
- [moa.template.provider.core \(module\), 136](#)
- [moa.template.template \(module\), 135](#)
- [moa.ui \(module\), 133](#)
- [moa.utils \(module\), 134](#)
- [moa.version \(module\), 135](#)
- [MoaCommandDoesNotExist, 129](#)
- [MoaDirNotWritable, 129](#)
- [moaDirOrExit\(\) \(in module moa.utils\), 134](#)
- [MoaFileError, 129](#)
- [MoaFormatter \(class in moa.logger\), 133](#)
- [MoaInvalidCommandLine, 129](#)
- [MoaPermissionDenied, 129](#)

## N

- [newJob\(\) \(in module moa.job\), 132](#)
- [newTestJob\(\) \(in module moa.job\), 132](#)
- [niceRunTime\(\) \(in module moa.utils\), 134](#)
- [NotAMoaDirectory, 130](#)
- [NotAMoaTemplate, 130](#)

## P

- [PolyYaco \(class in Yaco\), 137](#)
- [prepareExecute\(\) \(moa.job.Job method\), 131](#)
- [pretty\(\) \(Yaco.Yaco method\), 138](#)
- [printstack\(\) \(in module moa.utils\), 134](#)
- [private \(moa.jobConf.JobConf attribute\), 133](#)

## R

- [refresh\(\) \(in module moa.template\), 135](#)
- [refreshTemplate\(\) \(moa.job.Job method\), 131](#)
- [removeIndent\(\) \(in module moa.utils\), 134](#)
- [render\(\) \(in module moa.filesets\), 130](#)
- [resolve\(\) \(fist.fistCore method\), 140](#)
- [resolve\(\) \(fist.fistMapset method\), 141](#)
- [resolver\(\) \(fist.fistMapset method\), 141](#)
- [run\\_hook\(\) \(moa.job.Job method\), 132](#)

## S

- [save\(\) \(moa.jobConf.JobConf method\), 133](#)
- [save\(\) \(Yaco.Yaco method\), 138](#)
- [save\(\) \(Yaco.YacoDir method\), 139](#)
- [save\(\) \(Yaco.YacoFile method\), 139](#)
- [sendmail\(\) \(in module moa.utils\), 134](#)
- [setRecursiveVar\(\) \(moa.jobConf.JobConf method\), 133](#)

- [setTemplate\(\) \(moa.job.Job method\), 132](#)
- [simple\(\) \(Yaco.Yaco method\), 138](#)
- [simple\\_decorator\(\) \(in module moa.utils\), 135](#)
- [simpleRunner\(\) \(in module moa.actor\), 129](#)
- [soft\\_update\(\) \(Yaco.Yaco method\), 138](#)
- [sync\(\) \(in module moa.actor\), 129](#)

## T

- [Template \(class in moa.template.template\), 136](#)

## U

- [update\(\) \(Yaco.Yaco method\), 139](#)

## Y

- [Yaco \(class in Yaco\), 137](#)
- [Yaco \(module\), 136](#)
- [YacoDir \(class in Yaco\), 139](#)
- [YacoFile \(class in Yaco\), 139](#)