

# PhaME

"PhaME or **Phylogenetic and Molecular Evolution (PhaME)** is an analysis tool that performs phylogenetic and molecular evolutionary analysis."([Doc](#))

"Given a reference, PhaME extracts SNPs from complete genomes, draft genomes and/or reads, uses SNP multiple sequence alignment to construct a phylogenetic tree, and provides evolutionary analyses (genes under positive selection) using CDS SNPs."([Github](#))

As for the structure of PhaME: It is just a accumulation of several different tools into one command line tool that is configured with a text file of settings known as the **control file**.

## Resources

- [PhaME Documentation](#) - As a word of warning, the documentation for PhaME is **out of date and inconsistent in some places**.
- [PhaME Github](#) - As a word of warning: please ignore the readme, it says to install PhaME using Anaconda; Despite many attempts I never got this working using the Sep 23 2019 version . Please instead follow the installation steps found under [Installation with Docker Windows](#) instead.
- [Associated Paper](#)

## Installation with Docker (Windows)

**Note:** While these instructions don't specifically cover everything necessary for installing PhaME on machine not running Windows (i.e. Linux or Mac) none of the tools (Mostly, [docker desktop](#)) require Windows so with a little work it should run on any platform.

This is the easiest and most reliable installation method they have. That said some of their instruction are out of date so I've updated the [instructions on their guide](#) to use the latest version of PhaME as of 7/12/2022 below.

## Installing

**For Context:** Docker is a way of running software in a mini virtual machine. This can be convient for scaling applications accross many servers or in our case having a singular package which comes pre-packaged with all the 3rd party libraries/versions/etc all sorted. It also works cross platform.

1. Install docker from [here](#)
2. Download PhaME's image by running this command:

```
docker pull quay.io/biocontainers/phame:1.0.3--1
```

3. You can test that everything is installed and ready to use with the following command

```
docker run --rm quay.io/biocontainers/phame:1.0.3--1 bash -c 'phame --vcheck'
```

## Notes on attempts at other installation methods (and why not to use them if possible)

### Installing with Anaconda

This method will not work on Windows. [Anaconda](#) claims it works on Windows but if you try it the what feels like 1 in 10 times that Anaconda won't seemingly hang or run into some other unpredictable error it will say it can't find the

**pam1** package (presumably a dependency of the PhaME package). The reason for this is that according to [Anaconda](#), **pam1** doesn't support Windows so it can't find that package. Despite this the PhaME package still proports to support Windows on the Anaconda site.

This would mean it would be presumably possible to install on Linux but when I tried on [Ubuntu WSL](#) I was still plagued by issues with Anaconda hanging and running into errors constantly.

## Installing the Docker Web-interface

This is a way to install it on a production server and get email notifications for jobs finishing and more but it requires fixing some issues in the files and the whole thing is overkill for what were doing. Unless we wanted to use this tool repeatedly, often, and had a dedicated computer for it.

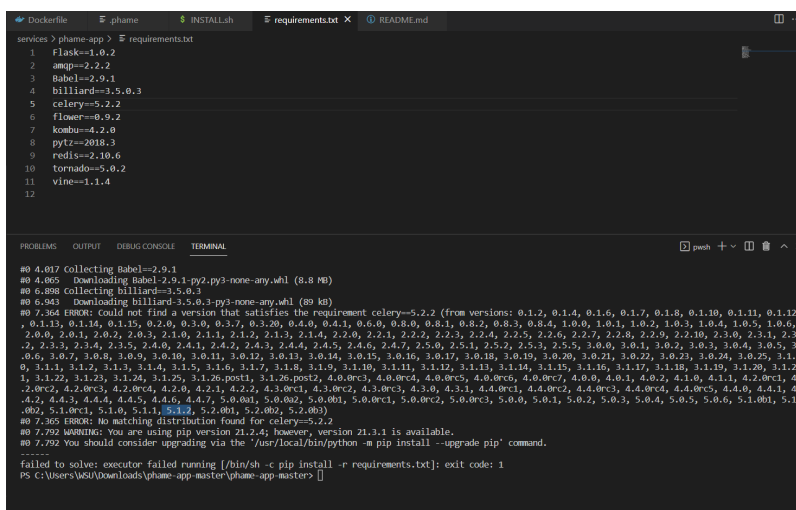
Disclaimer: it does not provide a fancy visual interface for setting the control file or anything for running PhaME. It just adds a visual interface to view many PhaME analysis jobs running, what resources of the comuter there using, etc.

The only reason these notes exist is that I mistakenly didn't try the normal [Installation with Docker Windows](#) method first. I suggest that way over this unless you really need those production server features.

1. First install docker from [here](#)
2. Download the phame web app repo from [here](#)
3. Follow the folling steps to fix the code which has broken versions.

The github auto update bot assigned celery to version 5.2.2 which doesn't seem to be valid so change that to something elset :

This needs to be changed in requirement files `services/phame-app/requirements.txt` and `services/celery-queue/requirements.txt`.



```
services > phame-app > requirements.txt
1  Flask==1.0.2
2  amqp==2.2.2
3  Babel==2.9.1
4  billiard==3.5.0.3
5  celery==5.2.2
6  flower==0.9.2
7  kombu==4.2.0
8  pytz==2018.3
9  redis==2.10.6
10 tornado==5.0.2
11 vine==1.1.4
12

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
no 4.007 Collecting Babel==2.9.1
no 4.005 Downloading Babel-2.9.1-py2.py3-none-any.whl (8.8 MB)
no 6.008 Collecting billiard==3.5.0.3
no 6.943 Downloading billiard-3.5.0.3-py3-none-any.whl (80 kB)
no 7.364 ERROR: Could not find a version that satisfies the requirement celery==5.2.2 (from versions: 0.1.2, 0.1.4, 0.1.6, 0.1.7, 0.1.8, 0.1.10, 0.1.11, 0.1.12, 0.1.13, 0.1.14, 0.1.15, 0.2.0, 0.3.0, 0.3.7, 0.3.20, 0.4.0, 0.4.1, 0.6.0, 0.8.0, 0.8.1, 0.8.2, 0.8.3, 0.8.4, 1.0.0, 1.0.1, 1.0.2, 1.0.3, 1.0.4, 1.0.5, 1.0.6, 2.0.0, 2.0.1, 2.0.2, 2.0.3, 2.1.0, 2.1.1, 2.1.2, 2.1.3, 2.1.4, 2.2.0, 2.2.1, 2.2.2, 2.2.3, 2.2.4, 2.2.5, 2.2.6, 2.2.7, 2.2.8, 2.2.9, 2.2.10, 2.3.0, 2.3.1, 2.3.2, 2.3.3, 2.3.4, 2.3.5, 2.4.0, 2.4.1, 2.4.2, 2.4.3, 2.4.4, 2.4.5, 2.4.6, 2.4.7, 2.5.0, 2.5.1, 2.5.2, 2.5.3, 2.5.5, 3.0.0, 3.0.1, 3.0.2, 3.0.3, 3.0.4, 3.0.5, 3.0.6, 3.0.7, 3.0.8, 3.0.9, 3.0.10, 3.0.11, 3.0.12, 3.0.13, 3.0.14, 3.0.15, 3.0.16, 3.0.17, 3.0.18, 3.0.19, 3.0.20, 3.0.21, 3.0.22, 3.0.23, 3.0.24, 3.0.25, 3.1.0, 3.1.1, 3.1.2, 3.1.3, 3.1.4, 3.1.5, 3.1.6, 3.1.7, 3.1.8, 3.1.9, 3.1.10, 3.1.11, 3.1.12, 3.1.13, 3.1.14, 3.1.15, 3.1.16, 3.1.17, 3.1.18, 3.1.19, 3.1.20, 3.1.21, 3.1.22, 3.1.23, 3.1.24, 3.1.25, 3.1.26.post1, 3.1.26.post2, 4.0.0rc1, 4.0.0rc2, 4.0.0rc3, 4.0.0rc4, 4.0.0rc5, 4.0.0rc6, 4.0.0rc7, 4.0.0, 4.0.1, 4.0.2, 4.1.0, 4.1.1, 4.2.0rc1, 4.2.0rc2, 4.2.0rc3, 4.2.0rc4, 4.2.0, 4.2.1, 4.2.2, 4.3.0rc1, 4.3.0rc2, 4.3.0rc3, 4.3.0, 4.3.1, 4.4.0rc1, 4.4.0rc2, 4.4.0rc3, 4.4.0rc4, 4.4.0rc5, 4.4.0, 4.4.1, 4.4.2, 4.4.3, 4.4.4, 4.4.5, 4.4.6, 4.4.7, 5.0.0a1, 5.0.0a2, 5.0.0b1, 5.0.0rc1, 5.0.0rc2, 5.0.0rc3, 5.0.0, 5.0.1, 5.0.2, 5.0.3, 5.0.4, 5.0.5, 5.0.6, 5.1.0b1, 5.1.0b2, 5.1.0rc1, 5.1.0, 5.1.1, 5.1.2, 5.2.0b1, 5.2.0b2, 5.2.0b3)
no 7.365 ERROR: No matching distribution found for celery==5.2.2
no 7.792 WARNING: You are using pip version 21.2.4; however, version 21.3.1 is available.
no 7.792 You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
failed to solve: executor failed running [/bin/sh -c pip install -r requirements.txt]: exit code: 1
PS C:\Users\BGO\Downloads\phame-app-master> phame-app-master>
```

I bumped celery=4.2.0. Presumably the github bot bumped it for a bug/security issue so ideally we would update this to something newer (it was some beta version of 4.2.0 before) but if it's just for a internal side project that is air gapped it is probably relatively safe. Also if you try and upgrade the celery version past 4.2.0 other packages cause errors because they to need to have version bumps and so on.

Next add to `services/phame/requirements.txt` the line, `email-validator==1.2.1`

Then in `services/phame/manage.py` add the following as a line at the beginning of the file: `#!/usr/bin/env python3`

At this point run the docker and check that all containers are running:

Create Containers: `docker-compose -f ./docker-compose-dev.yml build`

Start Docker: `docker-compose -f ./docker-compose-dev.yml up -d`

You can open the docker desktop app to view the containers and see what IPs they have websites on to navigate to in browser.

## The normal Docker install also works in WSL (Windows Linux Subsystem)

**(these notes are incomplete because I stopped after realizing it would work on Windows without WSL)**

This the easiest reliable way that I suggest but just through WSL as apposed to plain Windows.

On [Ubuntu WSL](#)(or any Debian based WSL probably) perform the following to install docker:

```
#https://docs.docker.com/engine/install/ubuntu/

sudo apt-get update

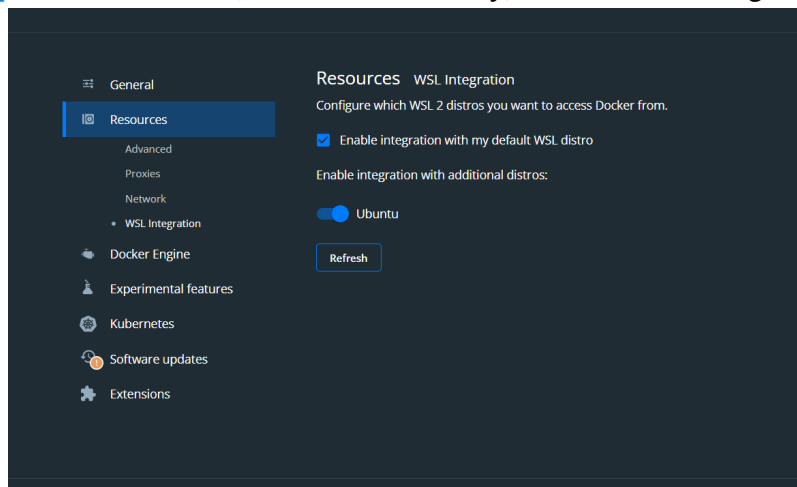
sudo apt-get install ca-certificates curl gnupg lsb-release

# install GPG key
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg

# configure repo
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee
/etc/apt/sources.list.d/docker.list > /dev/null

# install
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

Then open [docker desktop](#) over on Windows (install it if not already) and enable [wsl](#) integration:



From here install the PhaME image:

```
docker pull quay.io/biocontainers/phame:1.0.3--1
```

## How to run PhaME

PhaME expects a control file to be passed to it which in turn has some folder locations of the **workdir** and **refdir**.

To make running easier I've setup a **TEMPLATE folder** with a run script and all the needed folders in place. It can be found in the repository [here](#).

## Understanding the TEMPLATE folder

The folder called TEMPLATE folder is organized like so:

```
TEMPLATE/  
  refdir/  
  workdir/  
  output/ # phame results stored here  
  main.ctl # the control file  
  run.sh # script to run phame on this folder
```

It's meant to have everything setup so you can just copy the folder, rename it, **place files in it and run it with the run script**.

The **control file** (maint.ctl) in the template has a lot of settings you can play with. I've put in what I believe to be reasonable ones for our use case but by all means change them as you see fit. See the doc [here](#) for more about individual settings and what output files they produce.

The output files of PhaME are created under TEMPLATE/workdir/results but we move them in the run.sh to **output/timestamp** to make it easier to save results and run PhaME consecutively to try out different control files. Contained in the results is a tree folder where all of our trees will be stowed.

## Before running the first time on Windows

For running things in the commandline like PhaME I suggest using Gitbash on Windows so you can write shell scripts (Such as our **run.sh**) to automate running basic commands that are cross platform to Linux and Mac. Additionally, Gitbash comes with an install of Git so you often already have it a lot of the time.

In case you don't have it though, you can get Git and Gitbash [here](#). **This is necessary for your to run the shell script I've included on Windows (which itself isn't necessary but convient for running)**

Be advised there is unfortunately some Gitbash and docker compatibility issues. To operate on our files we need to mount them to the running docker container. [This doesn't work properly in gitbash \(and any shell scripts it runs\)](#). The work around is to add **MSYS\_NO\_PATHCONV=1** before **docker run**.

Example:

To run a container that just sits there and lets you terminal into it with a volume mounted.

```
MSYS_NO_PATHCONV=1 docker run -t -v $(pwd)/phame_examples/ecoli:/data  
quay.io/biocontainers/phame:1.0.3--0
```

## Which file types to use

- **For contigs** FASTA files seem to work well in testing.

- **For genomes** however, we may need to use `.gff` (General Feature Format) files over FASTA because they are necessary for the **cdsSNPS** option in the **control file** which gets forcibly enabled if you use the **PosSelect** option in the control file. The **PosSelect** option is where you select the molecular evolution analysis to perform. With it off (and in turn with **cdsSNPS** off) PhaME confusingly still produces tree files even though we are not running a evolutionary analysis? So I'm not sure if these trees are correct or not with PosSelect off. [See the doc for more information about this](#). This is regretablely as far as I got on this problem. I've kept PosSelect off in the TEMPLATE control file.

## Running

1. Copy the TEMPLATE folder elsewhere and rename it something.
2. Place any reference genomes you wish to use in the **refdir** folder
3. Place any contigs in the **workdir** folder. Importantly, rename the file extension of these files to `.contig`. This is how PhaME finds them and it will not work without them renamed.
4. Open Gitbash in folder your created in step 1. This can be done by opening that folder in file explorer and right clicking in the empty space of the window->more options->open gitbash
5. Run `./run.sh` It will take a long time to run so once it starts running NUCmer your safe to leave it and come back sometime later.
6. After it finishes go to output/`timestamp` to view the output PhaME created.

**Running PhaME consecutively in the same folder** is normally tricky because it requires you move the workdir/results folder before allowing you to run PhaME again. The `run.sh` automatically does this by moving everything into the **output** folder with a timestamp of the results so you can more easily run again. For some features though like continuing a job this may need to be disabled.

## Working with tree output files

The tree output files can be found under `output/<timestamp>/trees` after PhaME has run succesfully.

You can specify how to generate trees with the **tree** option in the **control file**.

PhaMes tree files are mostly in the [Newick Tree Format](#). The `show.r` file in TEMPLATE demonstraights how you can read this file format in, customize how it's displayed and plot it to the screen.

I also have some more example R code of customizing the look and how to plot a full tanglegram (which takes two philogenetic trees) over in the `tanglegramTesting` folder also found in the repo [here](#).