pythonanywhere

Debugging Playbook

Basic Checklist: Start Here!

What **Python version** do you expect the code to be running? `import sys; print(sys.version)`
If you use the wrong python at any stage, you may not be able to access the packages that you thought you installed. eg: Did you install packages with the right pip version? Or if you are using a virtualenv, did you install packages while the virtualenv was activated, and is the virtualenv python the expected python version?

Are you in the correct **working directory**? `import os; print(os.getcwd())`
Seeing errors such as \`*cannot import my_own_module*\`, or \`*cannot find myfile.txt*\`, or your sqlite database appears to be empty? Get `os.path.dirname(os.path.abspath(__file__)))`, and `sys.path.append()` it to import modules, or `os.path.join()` it with your relative paths to get absolute paths when referencing any files.

`Logging` tips:
- Make sure you are looking at the most updated logs- the newest is at the bottom.
- There are already logs written to `/var/www`. These should be the first things you check when there is an error.
  - You can also get links to specific log files from the PythonAnywhere dashboard tab
  - If your log file (eg: `server.log`) is empty it may have just been rotated after filling up. The old log is at `server.log.1`
- To add your own logging, just add a `print` statement, or use `logging` for timestamps and controlling what file you log to
  - If your print is buffered and not written to disk immediately, `import sys; sys.stdout.flush()`
  - For webapps, to log to error.log instead of server.log, use stderr instead of stdout, `print('hi', file=sys.stderr)`

Website Problems

The first thing to do is to check your website's **logs**!
If you are running Django, you should also run `./manage.py check`

My whole site is down

Run `python /var/www/your_wsgi.py` with the correct **python** and:

If you see any import errors or failures to find files or modules, you either have not installed the packages successfully to the correct **python**, or you have **working directory** problems.

If your wsgi.py never finishes running, you have code that blocks. eg: for flask apps, take out any `app.run()` or put them inside of a `if __name__ == '__main__'`.

After you get your wsgi.py to run without any errors, hit the reload button and see if your site is still down- if you had blocking code, sometimes just a reload is unable to clear out your old processes and you will have to email us to get us to clear it down for you.

Parts of my site don't work

Pick one single url that is consistently erroring, and start adding your own **logging** to that specific function to debug and find out what is going on.

If you have code that takes a long time to return a result, and sometimes it 50Xs and shows a \`*HARAKIRI*\` message in your serverlog, then you are running up against our hard 5 min timeout for web workers to respond.

If you intermittently see \`*OperationalError*\`s about MySQL server having gone away or having exceeded the max_user_connections, you need to configure your DB with the correct `pool_recycle` settings.

Be sure to reload your webapp to see your new changes.

Gotchas for Free Users

For free users, you only have 1 web worker. This means that if you require a response from a different endpoint on your site while you are loading a certain page, this will block forever.

Task Problems

If it seems like you are running the wrong **python** version, specify the python version explicitly in your task command.
eg: `python3.6 /path/to/my/command`

Console Problems

If your console to be killed with a \`*io limit exceeded*\` message, limit the amount it prints to the screen and redirect output to files by `command 2> stderr.log 1> stdout.log`

Notebook Problems

If you install a new python package, you may need to restart your notebook kernel to see it.

To use a notebook with a virtualenv, install `tornado==4.5.3` and `ipykernel==4.8.2` into your virtualenv and refresh your notebook page to see your virtualenv name available as a kernel to switch to.

To stop your notebook server completely, go to the Consoles tab, fetch all running processes, and kill all Jupyter (notebook) processes.

One magic function to automatically display charts instead of needing to manually display(): `%pylab inline`

Security & Access Problems

Use ssh tunneling to access your PythonAnywhere databases from outside of PythonAnywhere. eg: run this on your local machine:

```
ssh -L 3306:username.mysql.pythonanywhere-services.com:3306 username@ssh.pythonanywhere.com
```

If you are having trouble logging in to ssh or to your database:
- ssh is case sensitive, did you create your username with upper/lower case?
- Did you escape special characters correctly? Reset your password to something without unicode and special characters (\, /, *, . etc) and try again.

If you are using some program that repeatedly tries to login and fails, your account will be banned as a safety measure to avoid someone from brute-forcing your password. You should stop running that program and wait an hour before trying again.