ImmutableWorkstation Documentation

Release 0.0.1

Paul R Brian

Apr 05, 2019

Contents

CHAPTER 1

Intro

See README

CHAPTER 2

Getting Started

Firstly install the python (3) package on your machine:

```
sudo pip install immutableworkstation
```

First run:

```
$ immutableworkstation.py

Usage:
    immutableworkstation.py config
    immutableworkstation.py start (latest | next)
    immutableworkstation.py stop (latest | next)
    immutableworkstation.py login (latest | next)
    immutableworkstation.py buildDocker (latest | next)
    immutableworkstation.py makeDockerfile (latest | next)
    immutableworkstation.py status
    immutableworkstation.py quickstart
    immutableworkstation.py (-h | --help )

Options:
    -h --help    Show this screen
```

2.1 Configuration

Initially *immutableworkstation.py* can run without a config folder, but pretty much the only thing you can do is *quick-start* which will help setup a local config folder.

The example local config folder is stored in the python package and deployed to your machine as a data file. When we run *quickstart* we deploy that to '~.immutableworkstation' folder in your homedir

This has

- a config.ini file

- a *.latest* folder holding the templates to build "latest" Dockerfile
- a *.next* folder holding the templates to build the "next" Dockerfile

Config file is located as *~/.immutableworkstation/config.ini* It has following format and items

```
[default]
tagname    = workstation
instance_name =immutableworkstation
localhost   = 127.0.0.1
username    = pbrian
ssh_port    = 2222
devstation_config_root = ~/.immutableworkstation
terminal_command = /usr/bin/konsole -e
volumes = {"~/data":      "/var/data",
           "~/projects": "/var/projects",
           "~/secrets":  "/var/secrets:ro",
           "~/Dropbox":  "/var/Dropbox"
           }
```

volumes is a json-formatted string that will be converted during config reading. *tagname* is the tagname used to identify the docker *instance image_name* is the name used to identify the built docker image, from

which we will run an instance. You must build a docker instance.

localhost is obvious, probably needs to be removed *username* is the name of the (only?) user who will use the docker instance. As it is the only name and user that password is set to that as well.

***ssh_port* port for docker instance to listen on for ssh connections** from the host machine (how we talk to our dev machine)

devstation_config_root the location of the config file, plus other templates *terminal_command* - command to run before sshing to the running docker instance I am assuming you have *konsole*. if not adjust the config.

This will have files for the config ready to install - they will be place on '/usr/local/config' (TODO: rename that location to branded).:

```
$ immutableworkstation.py quickstart
```

You will be asked at least one question

2.2 Preparing a dockerfile

TBD

2.3 Getting started

1. Set up config - see above
2. Make a docker file
3. Build a Docker image

we are targetting windows, linux and apple machines so will need sensible simple scripts else the start up and try me out barrier will be too high. however having python scripts makes the development part waaay easier, and the templating is all in python anyhow, so I think we have to have some road bumps. I think anone wanting to try this out is going to be capable of installing py3 anyway. Our target demographic is developers who want more control.

I am building a one-stop shop developer machine on Docker which means it is a large Dockerfile - which is becoming unwieldy So I shall have a template folder, which will hold

dockerfile.skeleton This is the bones of the Dockerfile, with very simple replace-locations built in such as:

```
FROM ubuntu:18.04
ENV USERHOME /home/pbrian

{{ apt }}
^^^^^^^^^^^^^^^^^
this bit will get replaced with contents of `apt.template`
```

Constraints are that the {{ file }} must be on its own line, with only spaces between it and line start / end It is NOT using Jinja2, it just looks like it. Because one day it might.

Its that simple. We can play around with variables if we really need to.