

My Overkill Dotfiles Configuration (Featuring Automated Install and Test Pipelines)

13 Jan, 2019

[Skip to the interesting bits](#)

So you have a bunch of dotfiles (`.vimrc` , `.bashrc` , `.bash_profile`) that you've customized to your liking on your machine.

All Is Right, everything is configured perfectly.

But wait – Oh no! You have to use a new computer, and everything is all wrong! So you either move the dotfiles over, or just struggle with a Wrong configuration.

You decide to move the dotfiles over. You're trying to remember what config files were important, were they where even located, and then `scp` 'ing them over or some other hellish workflow.

Okay, this is annoying, but it works. You've move everything over.

Over time, you start making changes to your config on your new computer, and life is good.

Oh no! You have to go back to your old computer, and all your recent changes to your config aren't there! Oh god I don't even want to describe how painful resolving this one is going to be. Give up, there's no hope, all is lost.

If this is approximately you: you need a dotfiles repository!

Put your configuration files inside a repository on GitHub, use Dotbot, and install onto any new computer with just three commands:

```
git clone --recurse-submodules https://github.com/$YOUR_USERNAME/dotfiles.git
cd dotfiles
./install
```

If you're like me and switch jobs every eight months or often reformat your computers, some upfront investment in this is worth it!

There's a [lot of information](#) out there on this topic. I'm not going to provide a how-to, [other people have done that better than I can](#).

Rather, I'll implement the wisdom of this famous quote:

If you want to build a ~~ship~~ dotfiles configuration, don't drum up the men and women to gather ~~wood config files~~, divide the work, and ~~give orders~~ write a tutorial. Instead, teach them to yearn for the ~~vast and endless sea~~ convenience and status of a dotfiles setup.

What do I store in my dotfiles?

```
alex ~/.dotfiles$ l
azure-pipelines.yml  bin          gitconfig      Proun4b.jpg
bash_aliases         config       i3status.conf  README.md
bash_logout          dotbot       install        vimrc
bash_profile         fehbg       install.conf.json
bashrc               fzf         irssi
alex ~/.dotfiles$
```

.bash_aliases

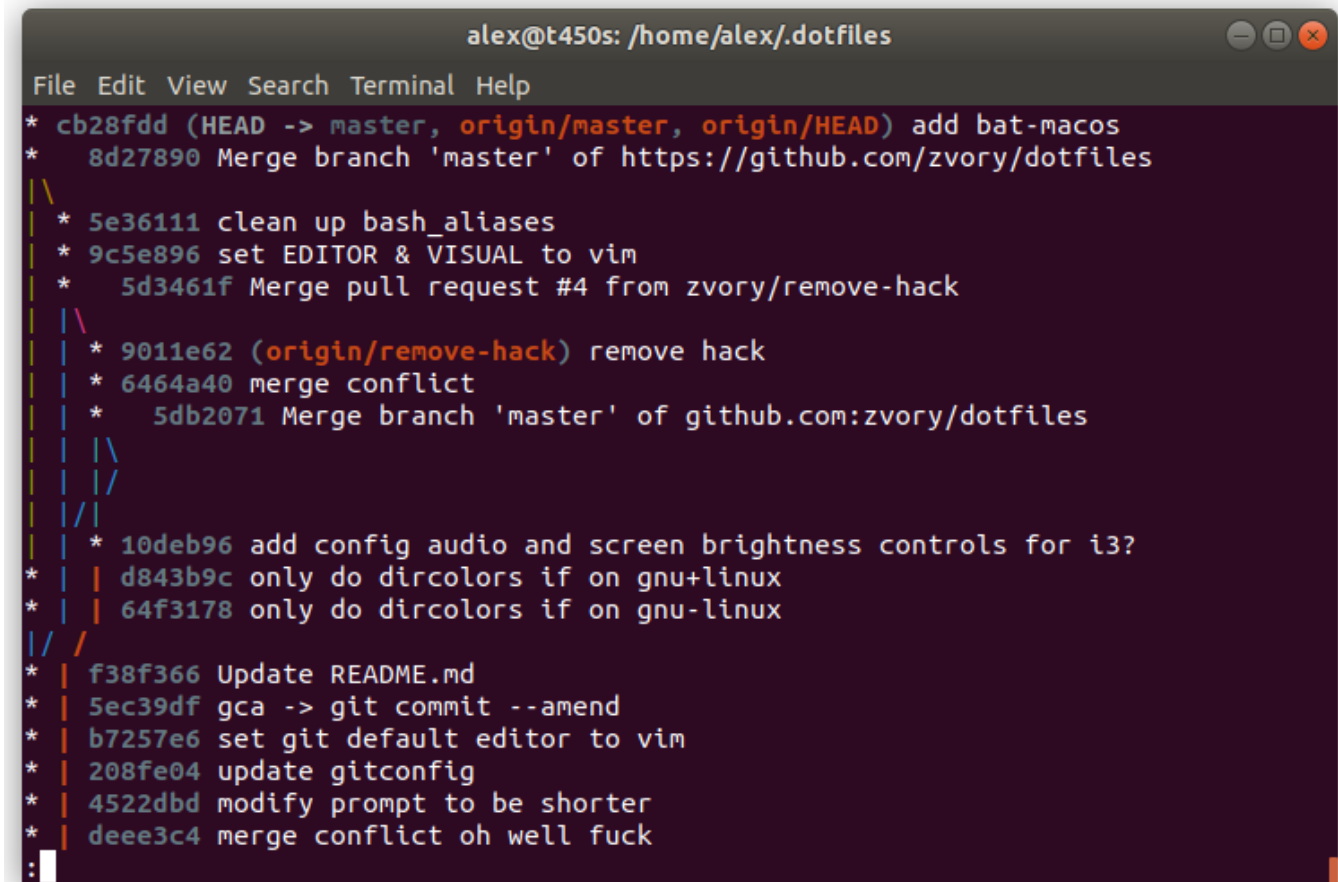
This contains a bunch of [useful aliases](#). Instead of having to type `ls`, I can just type `l`, or `la` instead of `ls -a`. Or typos I often make, like `alias chmoud=chmod`.

I have a bunch of [git aliases](#). e.g. `gcl` is `git clone` and `gco` is `git checkout`. These are *extremely* useful.

One pretty flashy git alias I have is `gll`:

```
alias gll='git log --graph --pretty=oneline --abbrev-commit'
```

which I use as a replacement for `git log`. It looks really nice:



```
alex@t450s: /home/alex/.dotfiles
File Edit View Search Terminal Help
* cb28fdd (HEAD -> master, origin/master, origin/HEAD) add bat-macos
* 8d27890 Merge branch 'master' of https://github.com/zvory/dotfiles
|
| * 5e36111 clean up bash_aliases
| * 9c5e896 set EDITOR & VISUAL to vim
| * 5d3461f Merge pull request #4 from zvory/remove-hack
|/
| * 9011e62 (origin/remove-hack) remove hack
| * 6464a40 merge conflict
| * 5db2071 Merge branch 'master' of github.com:zvory/dotfiles
|/
|/
| * 10deb96 add config audio and screen brightness controls for i3?
* | d843b9c only do dircolors if on gnu+linux
* | 64f3178 only do dircolors if on gnu-linux
|/
* | f38f366 Update README.md
* | 5ec39df gca -> git commit --amend
* | b7257e6 set git default editor to vim
* | 208fe04 update gitconfig
* | 4522dbd modify prompt to be shorter
* | deee3c4 merge conflict oh well fuck
:
```

Another thing I end up doing at every company is some alias like `cdchimera` -> `cd ~/company_name/code/rubycode/engine/chimera` so I can go straight to a project's directory.

I alias `cat` to `bat` and `ls` to `exa` which are basically just better versions of `cat` and `ls`.

```

▶ bat test.md
File test.md
1  # Markdown example
2
3  Note how we can correctly syntax-highlight the
4  code blocks *inside* the Markdown document.
5
6  Python example:
7
8  ``` python
9  cmd = "bat"
10 print("Hello from {}".format(cmd))
11 ```
12
13 Ruby example:
14
15 ``` ruby
16 cmd = "bat"
17 puts "Hello from #{cmd}"
18 ```

```

But `exa` and `bat` have different binaries for different OS's, and since I'm always switching between macOS and linux, I have to [switch on the OS](#) when aliasing.

```

if [[ "$OSTYPE" == "linux-gnu" ]]; then
    bat_binary="bat-linux"
    exa_binary="exa-linux"
elif [[ "$OSTYPE" == "darwin"* ]]; then
    bat_binary="bat-macos"
    exa_binary="exa-macos"
fi

```

Tip:

I'd often add an alias but never use it. A trick I use to get aliases into my muscle memory is to practice them.

For example: I made the alias `..` to execute `cd ..`, but I wasn't using it. When I realised this, I opened up a terminal, `cd` 'd into a directory, and typed `..` to go up one directory. And then I did this ten times. Just this amount of "practice" was enough to get me to use `..`.

binaries

I also have a bunch of [useful binaries](#) stored:

- `fzf` which is an *extremely* useful tool I recommend everyone use.
- The binaries for `exa` and `bat` (both linux and macOS builds)
- `ranger`, `diff-so-fancy`, and `fd` which I don't actually use these days, but are all allegedly great tools.

[My binaries are all in one folder](#) is linked to `~/bin` which is set to be in my `PATH` in my `bash_profile` so everything will work right out of the box.

.bashrc

I accumulated things over the years, lots of tiny small improvements like an `extract` function to handle the extraction of `.tar.bz2` / `.gz` / whatever files. I don't really know where most of the stuff came from.



misc

I also store my configurations for a number of other applications, like my `.gitconfig`, `i3config`, and such.

The install process

Dotbot

I use Dotbot. It's great. It's better than [GNU Stow](#), though I don't remember why. I also don't remember why it's better than all the other choices. Such is life.

Installation is as simple as simple as the three lines you saw at the beginning of this blog post, and looks like:

```
Creating link ~/.i3status.conf -> /home/vsts/.dotfiles/i3status.conf
Creating link ~/.bin -> /home/vsts/.dotfiles/bin
Creating link ~/.gitconfig -> /home/vsts/.dotfiles/gitconfig
Creating link ~/.bash_profile -> /home/vsts/.dotfiles/bash_profile
Creating link ~/.vimrc -> /home/vsts/.dotfiles/vimrc
Removing ~/.bashrc
Creating link ~/.bashrc -> /home/vsts/.dotfiles/bashrc
Removing ~/.bash_logout
Creating link ~/.bash_logout -> /home/vsts/.dotfiles/bash_logout
Creating directory /home/vsts/.config
Creating link ~/.config/i3 -> /home/vsts/.dotfiles/config/i3/
Creating link ~/.fehbg -> /home/vsts/.dotfiles/fehbg
Creating directory /home/vsts/Pictures
Creating link ~/Pictures/Proun4b.jpg -> /home/vsts/.dotfiles/Proun4b.jpg
Creating link ~/.irssi -> /home/vsts/.dotfiles/irssi/
Creating link ~/.bash_aliases -> /home/vsts/.dotfiles/bash_aliases
All links have been set up
Installing fzf [yes | ./fzf/install]
All commands have been executed
```






Azure Pipelines (build and test on Ubuntu and macOS)

I went to [GitHub Universe](#) in 2018. Microsoft had just recently acquired GitHub so there were a *lot* of booths advertising Microsoft products, especially [Azure](#). Specifically, they were advertising [Azure Pipelines](#), with a really incredible free tier. Ten parallel pipelines, unlimited minutes, for free on open source projects. I was like “wow!”, watched a demo on how you can run pipelines on three OS’s (Ubuntu, macOS, Windows) and was sold immediately.

So I tried to set it up so that I would “build” my dotfiles on **both macOS and Ubuntu** on every commit. That’s a pretty big deal! I don’t think Travis CI let’s you do this, and for dotfiles this is pretty critical!

Setting this up was pretty confusing,

But eventually I got it working:

Commit	Build #
 add bat-macos CI build for zvory	✓ 20190118.1
 Merge branch 'master' of https://github.com/zvory/dotfiles CI build for az-stripe	✓ 20190114.6
 clean up bash_aliases CI build for zvory	✓ 20190114.5
 set EDITOR & VISUAL to vim CI build for zvory	✓ 20190114.4
 Merge pull request #4 from zvory/remove-hack CI build for zvory	✓ 20190114.3

Now I have this cool badge on my repo: 

The builds

The builds are pretty simple.

The pipeline will clone my repo, and run the install script, then run `test.sh` to test that the aliases that are different based on operating system actually work. If any part fails, the build fails.

This is what a test looks like:

```
string="$(alias cat)" # string should look like "alias cat='bat-macos'"
if [[ $string == *"$bat_binary"* ]]; then #if string contains $bat_binary as a su
    echo "okay: bat alias"
else
    echo "err: bat alais"
    exit 1
fi
string="$(cat /dev/null)" #try to read from /dev/null
if [ $? -eq 0 ] # if error code is 0
then
    echo "okay: cat execution"
else
    echo "err: cat execution"
    exit 1
fi
```

(You can tell I don't know how to write bash scripts).

The pipelines themselves haven't been useful so far, but I definitely think it was a good idea for me to set them up:

- Getting your dotfiles to install and run in this cleanroom build environment makes you realise how many dependencies you actually have, forcing you to package them all up.
- The process of setting up the pipelines made my dotfiles a lot more agnostic to OS/resilient to weird edge cases.
- I think we're going to be on Unix-like systems for the next forty years, so I want to continuously evolve this dotfiles repo to make me more and more productive over time. It's a worthy investment. But this means my dotfiles will get ever more complex, and will eventually require this infrastructure. Maybe not in one year, but maybe in five or ten.

[You can see all of my builds here.](#)

Future improvements

- store iTerm configuration in here
- store Amethyst (macOS tiling window manager) config in here
- macOS settings, also <https://medium.com/@webprolific/getting-started-with-dotfiles-43c3602fd789>

Alex Zvorygin

Alex Zvorygin
samtciselseu@gmail.com



Software Engineer, Student.