Sh

If you're a regular shell user, you've almost ce
folder, which usually contains various tweaks,
directory to `$PATH`), telling your shell to do clever
aliases to commands (like `alias please=sudo`).

(If you're really organised, you'll have all your dotfiles in a rep
your settings synchronised across all the machines you work on.)

Anyhow, I suspect that few people know when things like `.bash_p`
executed. When I started, I just followed people's advice of putting stuf
didn't work, into `.bash_profile`. I could stop here and describe just the bash st
is), but there's a complication in that I switched to zsh a few years ago (and ha
occasionally use bash on machines which don't have zsh installed.

In order to handle this nicely then, I need to be able to specify things which are specific to
their own files, and then to specify things which any POSIX-compliant shell (like a
environment variables) can understand in a common startup file.

My solution to this problem is to define some new dotfile folders, one for each shell (`.bash/`, `.zsh/` a
`.sh/`), and one for the shell-independent files (`.shell/`):

```
.bash/
    env
    interactive
    login
    logout
.sh/
    env
    interactive
    login
.shell/
    env
    interactive
    login
    logout
.zsh/
    env
    interactive
```

"But!", y
kinds of sh

- [non-]i
- [non-]logi

All shells will first run
Once finished, login shells

Where to put st

It all depends on when it needs to be ru

If it's setting / modifying environment variab
environment variable (e.g., GREP_COLOR), it sh
umask set, and also define some useful functions
variables (like $PATH).

Even if you don't adopt anything else from my scheme, I'd re
functions are doing which differs from something like `export PAT`

That particular pattern is way too common, and is very dangerous if
(or whatever your variable is, like `$LD_LIBRARY_PATH`) isn't set. Then, the va
usually means both `/path/to/dir` *and the current directory*, which is usually b
and a security concern.

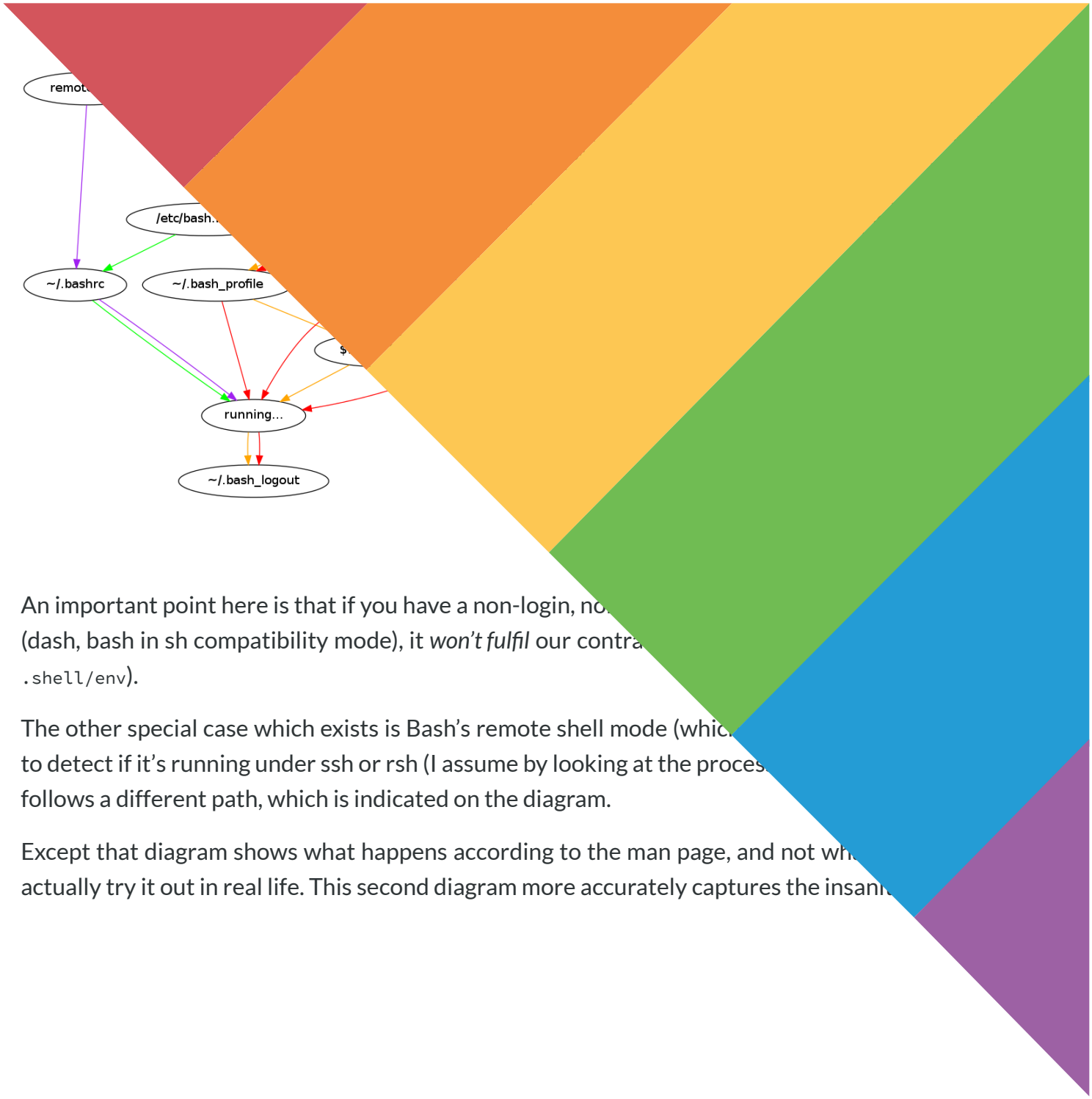
With my implementation (see `.shell/env_functions`), you can append, prepend and r
from any colon-separated environment variable, and when appending or prepen
guaranteed that directory will only then appear in that variable once.

As a side note, I'm very disappointed in my `indirect_expand()` function, so if you have a better so
please let me know (or send me a pull request).

Implementation

In order to implement this, you first need an understanding of which startup files are run in each case. Of course, this isn't hard, since all the shells have the same, sensible system. Ahahaha, no.

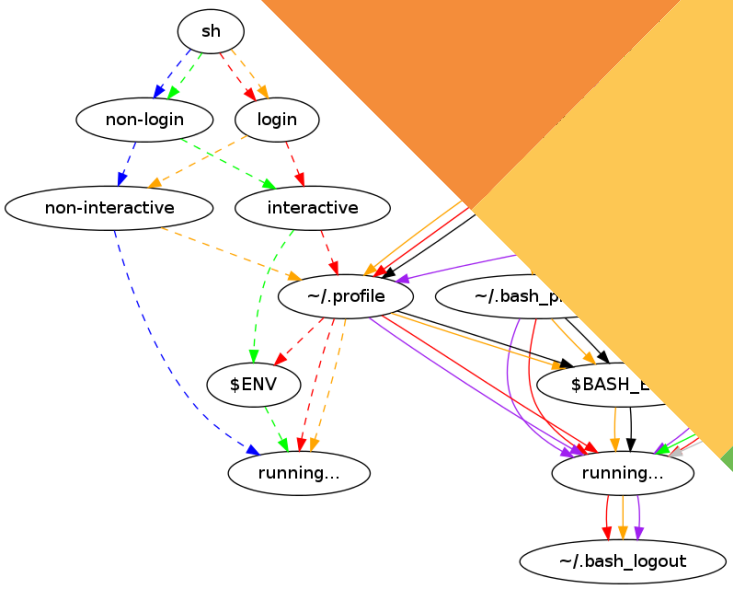
Fortunately, I've read the man pages for you, and drawn a pretty diagram. To read it, pick your shell, whether it's a login shell, whether it's interactive, and follow the same colour through the diagram. When the arrows split out to multiple files, it means that the shell will try to read each one in turn (working left to right), and will use the first one it can read.



An important point here is that if you have a non-login, no
(dash, bash in sh compatibility mode), it *won't fulfil* our contra
`.shell/env`).

The other special case which exists is Bash's remote shell mode (whic
to detect if it's running under ssh or rsh (I assume by looking at the proces
follows a different path, which is indicated on the diagram.

Except that diagram shows what happens according to the man page, and not wh
actually try it out in real life. This second diagram more accurately captures the insan



See how remote interactive login shells read /etc/bash.bashrc, bu
Sigh.

Finally, here's a repository containing my implementation and the graphvi
If your POSIX-compliant shell isn't listed here, or if I've made a horrible mi
please send me a pull request or make a comment below, and I'll update this post a
[1]and since I'm writing this, I can make you say whatever I want for the purposes of narrative.

Comments

Saxon 'Lemony' Landers • 7 years ago

Interesting, I've never really thought about setting variables for other shells than zsh, considering I use it almost exclusively.

she

But I thi

^ | v • Re

This comment was deleted.

flowblok Mod ➚ Guest • 7 year

The simplest example is a shel

I actually have no idea what happen
and let me know?

As for a definition of an interactive shell, here
An interactive shell is one started without non-op
whose standard input and error are both connected
isatty(3)), or one started with the -i option.
(and of course, all other shells are therefore non-interacti

^ | v • Reply • Share ›

Andrey • 3 years ago

Thank you very mush for the nice description and diagrams. But could you ple
with descriptions of arrows colors. It will greatly enhance their usability.

3 ^ | v • Reply • Share ›

Antonio Giráldez ➚ Andrey • 2 years ago

open the graphviz files: the legend you are looking for is in there

^ | v • Reply • Share ›

Shackra Seaslug • 4 months ago • edited

What do you do for stuff like zplug (which is something for zsh)?

EDIT: Okay, nvm, it is obvious that such zsh specific configuration can go into .zshrc. Next question: what if I want to leave the shell prompt as is, i.e.: for Emacs to work fine with Tramp with sudo.

EDIT 2: after organizing all my shells configurations I've been able to make Tramp work with sudo without further changes to my Emacs configuration, thanks!

1 ^ | v • Reply • Share ›

Adrian

6 years la

In my experienc
/etc/bash.bash_log
4.4/config-top.h:

```
/* System-wide .bashrc
/* #define SYS_BASHRC "

/* System-wide .bash_logout
/* #define SYS_BASH_LOGOUT "
```

(Yes, they're disabled by default.)

Check the *FILES* section of your system's bash ma

Reply • Share ›

Christian Herenz • 2 years ago

Hey, thanks for this nice summary. I encounterd that in openSUS
insanity to the whole issue, is that they source ~/.bashrc in /etc/profi
nuts and i filed a bug report, but I have little hopes that they will fix this:
https://bugzilla.opensuse.o...

The original logic beyond all the madness, is that you have fine grained control o
which case. If you follow the opensuse approach, then this all becomes meaningles
${HOME}/.bashrc is always evaluated.

Reply • Share ›

Aneesh P U • 7 years ago

Great post!

When I run bash as an interactive login shell, this seems to be the order of execution of the dotfiles and I think it is slightly different from what you have described.

/etc/bash.bashrc
/etc/profile
~/.bashrc
~/.profile

and when ~/. bash_profile is present
/etc/bash.bashrc
/etc/profile

Je

At the
diagrams
correct is th

So i did the test,
according, and the o

`~/.bash_profile || ~/.bash_

• Reply • Share ›

Ian Kirker ➚ flowblok • 7 years ago

To further complicate the issue, I believ
patches in the Debian build.

• Reply • Share ›

fumiyas • 7 years ago • edited

Where are /etc/zshrc and $ENV in the zsh diagram?

1 • Reply • Share ›

flowblok Mod ➚ fumiyas • 7 years ago

According to the man page, $ENV is not run in a normal zsh startup procedu
emulating sh).