Distributing loadable builtin bash modules

Asked 15 years, 9 months ago Modified 7 years, 5 months ago

Viewed 854 times

1

I've written a built-in for bash which modifies the 'cd' command, a requirement for my software. Is there a way to actually distribute a loadable independently of bash itself? I'd ideally like to distribute just a drop in "additional feature" because I know people can be put off by patching and compiling their shell from source code.

I want to time how long a user is in a directory so I can determine where they want to be. It's this functionality: http://github.com/joelthelion/autojump/tree/master rewritten as a bash builtin, for performance issues. This implementation uses `$PROMPT_COMMAND` to work but I wanted something integrated.

c bash cd distribute built-in

Share

Improve this question

Follow

edited Jul 22, 2017 at 2:09

avpaderno

29.6k ● 17 ● 78 ● 94

asked Mar 14, 2009 at 14:47

Philluminati

2,789 ● 2 ● 26 ● 34

2 Answers

Sorted by: Highest score (default)

3

It is unclear what you have modified but in any case, `bash` (like at least `ksh93` which IIRC introduced the concept and `zsh`) supports, using the `enable -f file name` syntax, loading built-in functions as external dynamically loaded modules.

These modules being plain files can certainly be distributed independently, as long as you make sure they are compatible with the target version/architecture. This was already true 5 years ago when you asked this question.

One issue in your case is there seems to be no documented way to overload a internal built-in like `cd` by a dynamically loaded one while keeping the ability to access the former.

A simple workaround would be to implement your customized cd with a different name, say mycd, like this:

```
int mycd_builtin(list)
WORD_LIST *list;
{
  int rv;
  rv=cd_builtin(list);
  if(rv == EXECUTION_SUCCESS) {
    char wd[PATH_MAX+1];
    getcwd(wd,sizeof(wd));
    // do your custom stuff knowing the new working di
    ...
```

```
  }
  return (rv);
}
```

then to use an alias, or better, a shell function for your customized version to be used instead of the regular one:

```
cd() {
  mycd "$@"
}
```

As long as your customization doesn't affect the behavior of the standard command and thus doesn't risk breaking scripts using it, there is nothing wrong in your approach.

Share Improve this answer

Follow

answered Sep 21, 2013 at 13:06

jlliagre

30.8k ● 7 ● 64 ● 74

A bit of counter-googling and actually looking doing 'help enable' in my shell leads me to conclude you are indeed correct. – Philluminati Oct 2, 2013 at 14:30

Thank you for taking the time to respond to this. – Philluminati Oct 2, 2013 at 14:31

2 It took me four years ;-) – jlliagre Oct 2, 2013 at 19:54

2

Changing the built-in cd is a support nightmare for any admin and unwelcome to foreign users. What is wrong with naming it 'smart-cd' and letting the USER decide if they want the functionality by including it in their .bashrc

or .profile? Then they can setup things however they want.

Also, using how long you've been in a directory is a pretty poor indication of preference. How would you distinguish between idling (a forgotten shell hanging in /tmp overnight), long-running scripts (nightly cron jobs), and actual activity.

There are a multitude of other methods for creating shortcuts to favorite directories: aliases, softlinks, $VARIABLES, scripts. It is arrogant of you to assume that your usage patterns will be welcomed by other users of your system.

answered Mar 15, 2009 at 2:50

HUAGHAGUAH
1,071 ● 6 ● 3