



Stack Overflow is a question and answer site for professional and enthusiast programmers. It's 100% free, no registration required.

Tell me more x

## Accesssing bash completions for specific commands programmatically

We already help you answer all your questions.  
Now let us help you find your next coworker. 

I'm trying to write a small command launcher application, and would like to use bash's tab completions in my own completion system. I've been able to get a list of completions for general commands using `compgen -abck`.

However, I would also like to get completions for specific commands: for instance, the input `git p` should display completion for git's commands.

Is there any way I can use `compgen` to do this? If not, are there any other ways I can get a list of completions programmatically?


Thanks for your help.

[EDIT: To clarify, I'm not trying to **provide** completion to bash - my app is a GUI command launcher. I'd simply like to use bash's existing completions in my own app.]

`bash` `shell` `tab-completion` `bash-completion`

edited Sep 3 '10 at 22:44

asked Aug 19 '10 at 10:41

 **Donald Harvey**  
892 5 16

## 5 Answers

I don't really know how it works, but the awesome window manager uses the following lua code for getting access to bash completion's result:

[http://git.naquadah.org/?](http://git.naquadah.org/?p=awesome.git;a=blob;f=lib/awful/completion.lua;h=db0391d73e9ebe5427971fe731dfbae44652dc45;hb=HEAD#l107)

`p=awesome.git;a=blob;f=lib/awful/completion.lua;h=db0391d73e9ebe5427971fe731dfbae44652dc45;hb=HEAD#l107`

*Edit:* Bash completion for "git l".

1. Via "complete -p" we find "complete -o bashdefault -o default -o nospace -F \_git git". We remember "\_git" for later.
2. The length of "git l" is 5, so we set COMP\_COUNT=6. We are completing the first argument to "git", so COMP\_CWORD=1.
3. All together we use the following command:

```
source /etc/bash_completion
__print_completions() {
  for ((i=0;i<${#COMPREPLY[*]};i++))
  do
    echo ${COMPREPLY[i]}
  done
}
COMP_WORDS=(git l)
COMP_LINE="git l"
COMP_COUNT=6
COMP_CWORD=1
_git
__print_completions
```

Output: "loa"

edited Sep 4 '10 at 9:25

answered Sep 3 '10 at 22:52

Uli Schlachter  
1,519 ●5 ●14

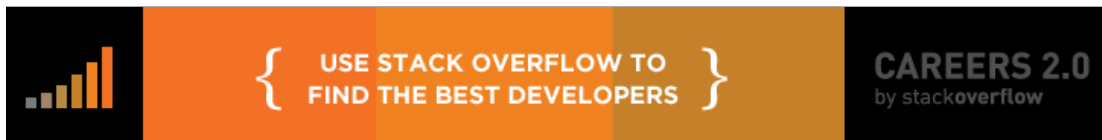
The code appears to do pretty much exactly what I want to do with my app. I'll need to port it to regular Bash, but it's still very helpful. Thanks a lot! – [Donald Harvey](#) Sep 3 '10 at 23:11

I did some testing and this is what is run to complete "git I": `/usr/bin/env bash -c 'source /etc/bash_completion; __print_completions() { for ((i=0;i<${#COMP_REPLY[*]};i++)); do echo ${COMP_REPLY[i]}; done }; COMP_WORDS=(git I); COMP_LINE="git I"; COMP_COUNT=6; COMP_CWORD=1; __git; __print_completions'` – [Uli Schlachter](#) Sep 4 '10 at 8:52

"\_git" is found via "complete -p" which prints "complete -o bashdefault -o default -o nospace -F \_git git". So if the first word is "git", call "\_git". COMP\_COUNT is the length of the string so far plus one. COMP\_CWORD is the word to complete (first one is index 0). – [Uli Schlachter](#) Sep 4 '10 at 8:54

Thanks, this'll be perfect. I was thinking of answering the question myself to provide the Bash code and abstract way of doing this, but you've done it for me :P. Anyway, I've awarded you the bounty - thanks again for your help! – [Donald Harvey](#) Sep 4 '10 at 9:45

- Hey great! Thanks for working this out. Note that for the sake of readability/convenience, you can replace the `for ((...;...;...))` statement with `for reply in "${COMP_REPLY[@]}"` (and then `echo "$reply"` inside the loop). Also: `i` in your example, or `reply` in my proposal, should be declared as `local` to avoid clobbering identically named globals. – [intuited](#) Apr 17 '11 at 22:18



Take a look at the Debian Administration [An introduction to bash completion](#) - it's fairly easy to understand - by the way, it also has a [part two](#).

answered Aug 29 '10 at 19:38

bruno nery  
682 ●4 ●12

This seems like a decent guide to writing functions that *provide* completion sets for commands. However, the OP is looking for a way to *invoke* that functionality programmatically, in the same way that bash itself does when the user hits the `TAB` key. – [intuited](#) Apr 17 '11 at 22:01

- PS: ordinarily I wouldn't downvote something like this, because it does provide useful info, but at the time it was at 4 points and the actual answer to the question was at 1. Hopefully this way people will be less likely to spend time reading through the linked page looking for an answer that isn't there. – [intuited](#) Apr 17 '11 at 22:20

Check in the `/etc/bash_completion.d/` directory. This is where the different command completion scripts stay.

answered Aug 22 '10 at 16:19

Daniel Velkov  
7,351 ●2 ●24 ●46

I have had a poke around in `/etc/bash_completion.d/`, but haven't yet worked out an easy way to get the completions contained within yet very easily. (I'm using Python to access the completions, making things slightly more complicated.) I was hoping there might be a simpler way of accessing command-specific completions. – [Donald Harvey](#) Aug 28 '10 at 18:40

Here a simple but working example in bash :

```
function foo_completion()
{
    local currentWord=${COMP_WORDS[COMP_CWORD]}
    local completionList=""

    case "${COMP_CWORD}" in
        "1")
            completionList="command1 command2 command3";
            ;;
        "2")
            completionList="param1 param2 param3";
            ;;
    esac

    COMPREPLY=( $( compgen -W "${completionList}" -- ${currentWord} ) )
}

complete -F foo_completion foo
```

With this kind of code, you will get commandN completed when you type "foo c" + tab and paramN completed when you type "foo command1 p" + tab

You can compute the completion list from the command help.

my2c

answered Aug 30 '10 at 8:08



neuro

5,578 ● 1 ● 11 ● 30

---

I hacked up this script a while back which gives you pretty accurate [Man-page Bash Completion](#) I know other people have done similar things with parsing `--help` option output.

answered Sep 3 '10 at 19:25



David Blevins

8,978 ● 19 ● 42

---

Great script, but not quite what I'm trying to achieve. Thanks anyway for linking to it though. – [Donald Harvey](#)

Sep 3 '10 at 22:51

---

---

Not the answer you're looking for? Browse other questions tagged [bash](#) [shell](#)

[tab-completion](#) [bash-completion](#) or [ask your own question](#).