

Forum:[Hack Forums](#)

Thread: [\[Bash Script\] Go back to menu, when an option is selected](#)

friiz1337 - November 7, 2019, 11:09 am

Hello guys

Down here I have a code that lets a user login into a system, as well as create an account. My question is: How can I loop this same code every time the user creates an account (chooses option 2, specifies username and password, and then shows the same menu(in case the user wants to create another account)).

I know bash doesn't have a goto option so I tried wrapping this whole piece of code into a function, but without luck (I miss something) . :/

Thanks for the help.

Code echo

```
echo
welcomeMessage          #welcome message function
echo
echo 1: Login            #login option
echo 2: Create account   #create account option
echo
echo Enter your option:
read options              #read option
case $options in
    "1")                  #login
        echo Enter Username
        read username      #read username entered
        grabUser           #function to check if username exists and return 1 if he exists
        if [ "$userResult" == 1 ]; then
            enterPass       #prompt password field

        else
            userResult=0
            while [ "$userResult" == 0 ]; do
                echo -n Try again:
                read username
                grabUser
            done
            enterPass
        fi
        grabPass           #function to check if password exists and return 1 if it exists
        if [ "$passResult" == 1 ]; then
            echo
            echo
            welcomeASCII
        else
            passResult=0
            while [ "$passResult" == 0 ]; do
```

```

        echo Try again:
        read -s password
        grabPass
    done
    echo
    echo
    welcomeASCII          #welcome screen ASCII function
fi
echo
echo
;;
"2")                    #create user option
    echo Create Username:
    read username
    grabUser
    while true; do      #while grabUser grabs username
        if [ "$userResult" == 0 ]; then      #if username doesn't exist
            insertUser          #add user to db
enterPass          #prompt password field
            insertPass          #add password to db
        else
            userResult=1
            while [ "$userResult" == 1 ]; do #loop until user doesn't exist
                echo Try Again
                read username
                grabUser
            done
            insertUser
            enterPass
            insertPass
        fi
    done
esac

```

m3rkill - November 8, 2019, 9:58 pm

how about making the script start itself again instead of quit?

netWalker - November 12, 2019, 1:52 pm

Here's how I'd do it.

```

Code #!/bin/bash
make_account() {
    do stuff...
    do more stuff...
}

```

```

while true
do
    make_account
done

```

Of course you could also just make the entire thing a while loop. Something like this:

```

Code #!/bin/bash
while true

```

```
do
  make account commands...
  make account commands....
done
```

A few side notes unrelated to your question. Do what you want with this, just my opinion.

1) The "echo" commands

There's an unnecessary number of them. It makes it not very pretty to read. I'm not judging, I used to do the same thing. Instead you should use something like this: `Code echo -e "Make account\n\n"` It's the same thing as `Code echo "Make account"`

```
echo
```

`echo` But it calls `echo` less, which makes things marginally faster. And it's easier on the eyes.

2) Make a prompt for "read."

You can give the `read` command a prompt by using this: `read -p "Select an option: " option`. Just a cosmetic thing really. Purely optional.

3) Might as well make that "case" statement an if-else. You really only have 3 possible options. Login, make account, and wtf ever else the retarded user could put in. This is again a personal preference thing, but it just seems a bit unnecessary. You could just make it a if-else.

```
Code #!/bin/bash
```

```
if [ $option == 1 ]; then
```

```
  code...
```

```
  code...
```

```
elif [ $option == 2 ]; then
```

```
  code...
```

```
  code...
```

```
else
```

```
  echo "$option is an invalid option"
```

```
fi
```

friiz1337 - November 13, 2019, 9:26 am

(Nov 12, 2019 02:52 PM)netWalker Wrote: [\[CITE\]](#) Here's how I'd do it.

```
Code #!/bin/bash
```

```
make_account() {
```

```
  do stuff...
```

```
  do more stuff...
```

```
}
```

```
while true
```

```
do
```

```
  make_account
```

```
done
```

Of course you could also just make the entire thing a while loop. Something like this:

```
Code #!/bin/bash
```

```
while true
```

```
do
```

```
  make account commands...
```

```
  make account commands....
```

```
done
```

A few side notes unrelated to your question. Do what you want with this, just my opinion.

1) The "echo" commands

There's an unnecessary number of them. It makes it not very pretty to read. I'm not judging, I used to do the same thing. Instead you should use something like this:

```
Code echo -e "Make account\n\n"
Code echo "Make account"
echo
```

echo But it calls echo less, which makes things marginally faster. And it's easier on the eyes.

2) Make a prompt for "read."

You can give the read command a prompt by using this: `read -p "Select an option: " option`. Just a cosmetic thing really. Purely optional.

3) Might as well make that "case" statement an if-else. You really only have 3 possible options. Login, make account, and wtf ever else the retarded user could put in. This is again a personal preference thing, but it just seems a bit unnecessary. You could just make it a if-else.

```
Code #!/bin/bash
if [ $option == 1 ]; then
    code...
    code...
elif [ $option == 2 ]; then
    code...
    code...
else
    echo "$option is an invalid option"
fi
```

Wow, thanks for the great answer, I will modify my code and get back to this thread. As for the tips, thank you, I'm sort of new to programming so anything is helpful.

netWalker - November 14, 2019, 9:29 am

(Nov 13, 2019 10:26 AM)friiz1337 Wrote: [\[CITE\]](#) Wow, thanks for the great answer, I will modify my code and get back to this thread. As for the tips, thank you, I'm sort of new to programming so anything is helpful.

Glad I could help. Good luck with your script.

friiz1337 - January 11, 2020, 2:34 am

So, it's been a while since my last reply, however I managed to solve my problem. I just used the `exec "$0"` command right after my "register" option loop. This allows the program to restart without creating additional processes, after the user has created an account. I also changed the case with if-elif's.

Another option that was pretty close to solving my problem was using `break 2/3` (depending on your loops).

Thank you for the help.

FantaŠec - February 3, 2020, 6:17 pm

If you're still looking to improve this, can you post an updated version? You shouldn't have to use `exec` at all to leverage this functionality (`exec` isn't usually ideal for a case like this too).

AnonMouse - February 6, 2020, 6:24 am

this would be cool to use if you could make it smaller

friiz1337 - February 6, 2020, 1:37 pm

(Jan 11, 2020 03:34 AM)friiz1337 Wrote: [\[CITE\]](#) So, it's been a while since my last reply, however I managed to solve my problem. I just used the `exec "$0"` command right after my "register" option loop. This allows the program to restart without creating additional processes, after the user has created an account. I also changed the case with if-elif's.

Another option that was pretty close to solving my problem was using `break 2/3` (depending on your

loops).

Thank you for the help.

I actually updated this by using functions lol. Calling a function works like a charm