

Linux

Expect command and how to automate shell scripts like magic

📅 2017-02-27 👤 admin 💬 0 Comments

In the previous post, we talked about writing **practical shell scripts** and we saw how it is easy to write a shell script. Today we are going to talk about a tool that does magic to our shell scripts, that tool is the **Expect command** or **Expect scripting language**.

Expect command or expect scripting language is a language that talks with your interactive programs or scripts that require user interaction.

Advertisements

Expect scripting language works by expecting input, then the Expect script will send the response without any user interaction.

You can say that this tool is your robot which will automate your scripts.

If Expect command is not installed on your system, you can install it using the following command:

```
$ apt-get install expect
```

Or on Red Hat based systems like CentOS:

```
$ yum install expect
```

Table of Contents [hide]

- 1 Expect Command**
- 2 Using autoexpect**
- 3 Working with Variables**
- 4 Conditional Tests**
- 5 If else Conditions**
- 6 While Loops**
- 7 For Loops**
- 8 User-defined Functions**
- 9 Interact Command**

Expect Command

Before we talk about expect command, Let's see some of the expect command which used for interaction:

- | | |
|----------|---|
| spawn | Starting a script or a program. |
| expect | Waiting for program output. |
| send | Sending a reply to your program. |
| interact | Allowing you in interact with your program. |

- The spawn command is used to start a script or a program like the shell, **FTP**, Telnet, SSH, SCP, and so on.
- The send command is used to send a reply to a script or a program.
- The Expect command waits for input.
- The interact command allows you to define a predefined user interaction.

We are going to type a shell script that asks some questions and we will make an Expect script that will answer those questions.

First, the shell script will look like this:

```
#!/bin/bash
echo "Hello, who are you?"
read $REPLY
echo "Can I ask you some questions?"
read $REPLY
echo "What is your favorite topic?"
read $REPLY
```

Now we will write the Expect scripts that will answer this automatically:

```
#!/usr/bin/expect -f
set timeout -1
spawn ./questions
expect "Hello, who are you?\r"
```

```
send -- "Im Adam\r"
expect "Can I ask you some questions?\r"
send -- "Sure\r"
expect "What is your favorite topic?\r"
send -- "Technology\r"
expect eof
```

The first line defines the expect command path which is `#!/usr/bin/expect` .

On the second line of code, we disable the timeout. Then start our script using spawn command.

We can use spawn to run any program we want or any other interactive script.

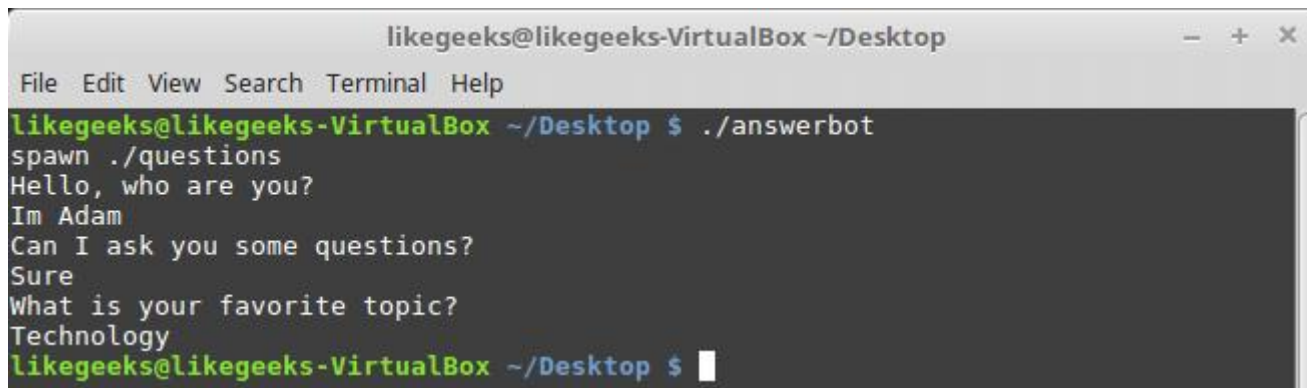
The remaining lines are the Expect script that interacts with our shell script.

The last line if the end of file which means the end of the interaction.

Now Showtime, let's run our answer bot and make sure you make it executable.

```
$ chmod +x ./answerbot
```

```
$ ./answerbot
```

A screenshot of a terminal window titled 'likegeeks@likegeeks-VirtualBox ~/Desktop'. The window has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal shows a prompt 'likegeeks@likegeeks-VirtualBox ~/Desktop \$' followed by the command './answerbot'. The script then prints 'spawn ./questions', 'Hello, who are you?', 'Im Adam', 'Can I ask you some questions?', 'Sure', 'What is your favorite topic?', and 'Technology'. The prompt returns to 'likegeeks@likegeeks-VirtualBox ~/Desktop \$' with a cursor.

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ ./answerbot
spawn ./questions
Hello, who are you?
Im Adam
Can I ask you some questions?
Sure
What is your favorite topic?
Technology
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Cool!! All questions are answered as we expect.

If you get errors about the location of Expect command you can get the location using the which command:

```
$ which expect
```

We did not interact with our script at all, the Expect program do the job for us.

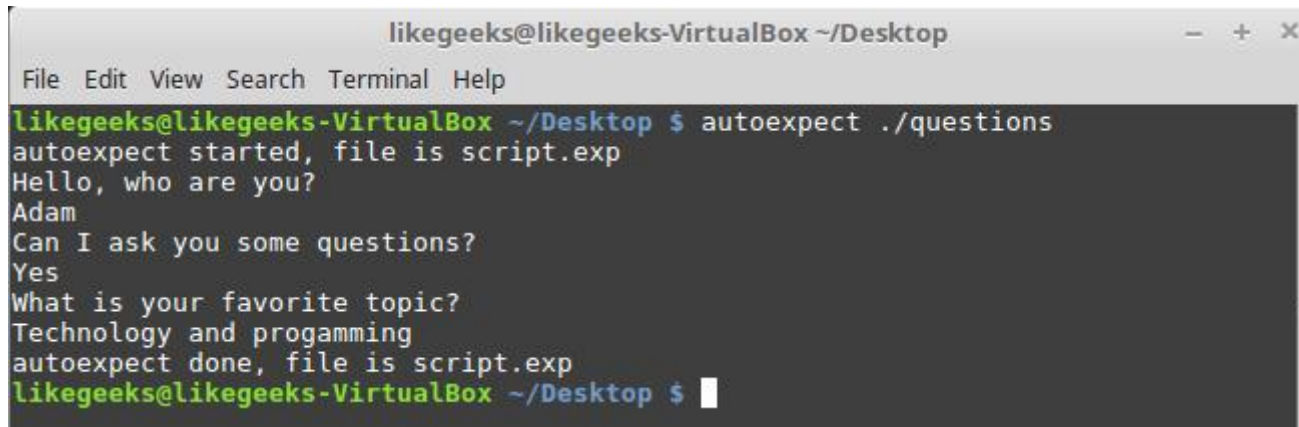
The above method can be applied to any interactive script or program. Although the above Expect script is very easy to write, maybe the Expect script little tricky for some people, well you have it.

Using autoexpect

To build an expect script automatically, you can use the autoexpect command.

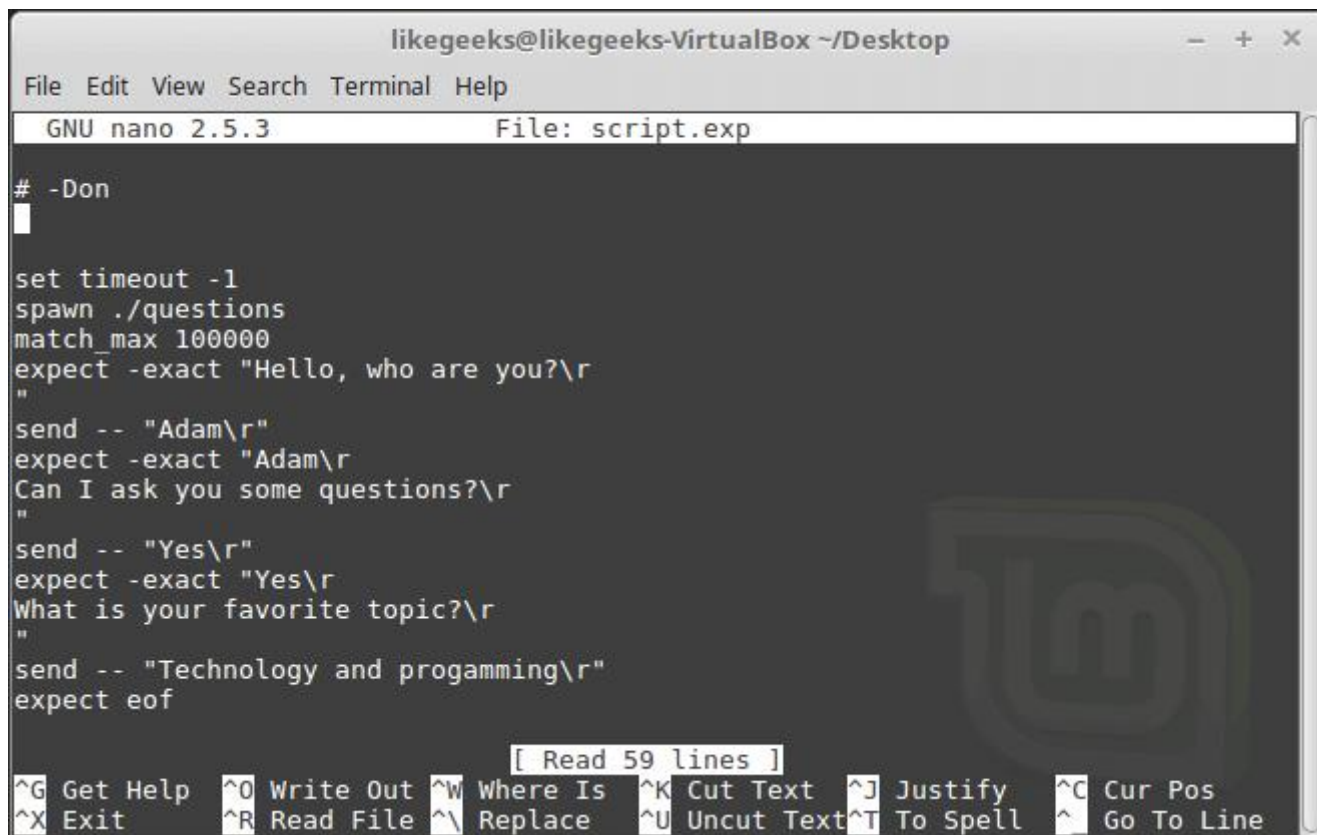
autoexpect works like expect, but it builds the automation script for you. The script you want to automate is passed to autoexpect as a parameter and you answer the questions and your answers are saved in a file.

```
$ autoexpect ./questions
```



```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ autoexpect ./questions
autoexpect started, file is script.exp
Hello, who are you?
Adam
Can I ask you some questions?
Yes
What is your favorite topic?
Technology and programming
autoexpect done, file is script.exp
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

A file is generated called script.exp contains the same code as we did above with some additions that we will leave it for now.



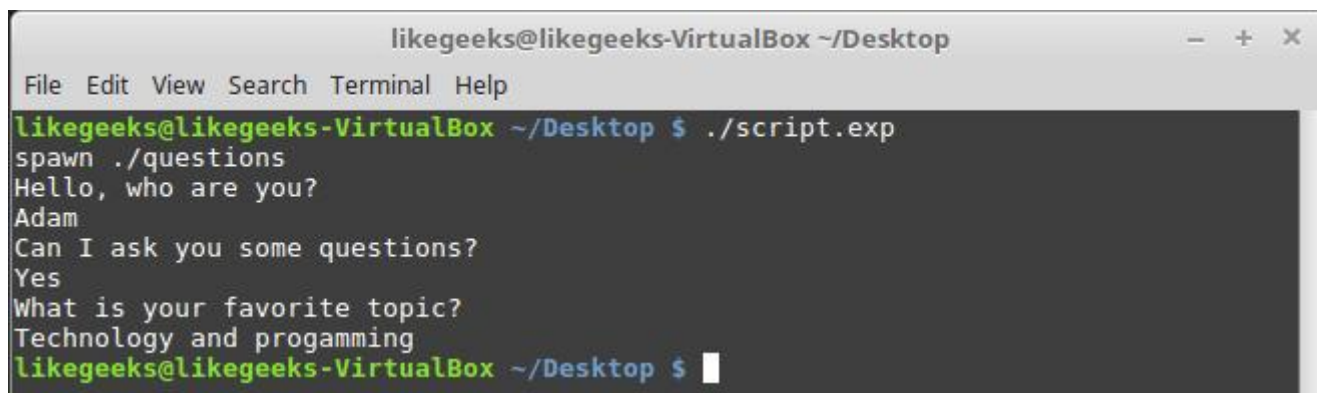
```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
GNU nano 2.5.3 File: script.exp

# -Don

set timeout -1
spawn ./questions
match_max 100000
expect -exact "Hello, who are you?\r"
"
send -- "Adam\r"
expect -exact "Adam\r"
Can I ask you some questions?\r
"
send -- "Yes\r"
expect -exact "Yes\r"
What is your favorite topic?\r
"
send -- "Technology and programming\r"
expect eof

[ Read 59 lines ]
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^_ Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

If you run the auto generated file script.exp, you will see the same answers as expected:



```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ ./script.exp
spawn ./questions
Hello, who are you?
Adam
Can I ask you some questions?
Yes
What is your favorite topic?
Technology and programming
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Awesome!! That super easy.

There are many commands that produce changeable output, like the case of FTP programs, the expect script may fail or stuck. To solve this problem, you can use wildcards for the changeable data to make your script more flexible.

Advertisements

Working with Variables

The set command is used to define variables in Expect scripts like this:

```
set MYVAR 5
```

To access the variable, precede it with \$ like this \$VAR1

To define command line arguments in Expect scripts, we use the following syntax:

```
set MYVAR [lindex $argv 0]
```

Here we define a variable MYVAR which equals the first passed argument.

You can get the first and the second arguments and store them in variables like this:

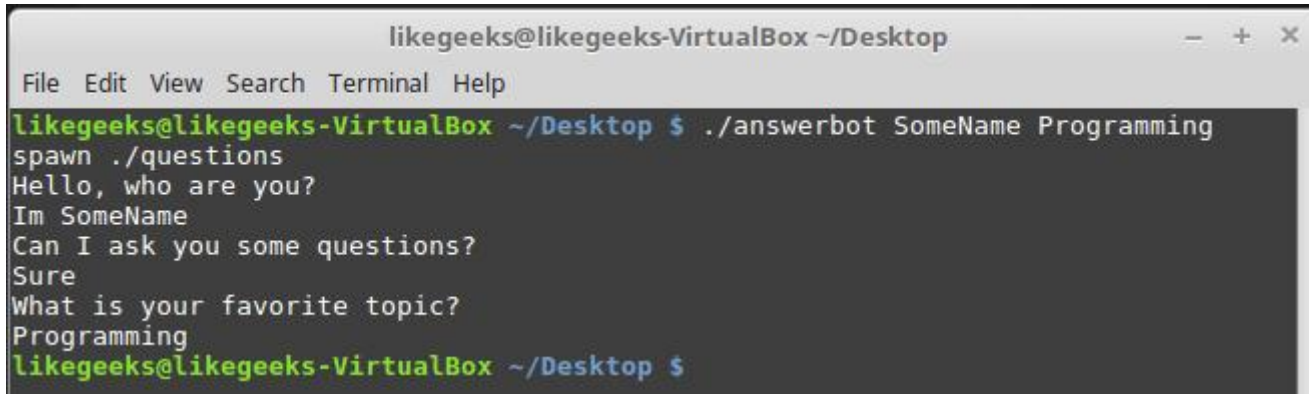
```
set my_name [lindex $argv 0]  
set my_favorite [lindex $argv 1]
```

Let's add variables to our script:

```
#!/usr/bin/expect -f
set my_name [lindex $argv 0]
set my_favorite [lindex $argv 1]
set timeout -1
spawn ./questions
expect "Hello, who are you?\r"
send -- "Im $my_name\r"
expect "Can I ask you some questions?\r"
send -- "Sure\r"
expect "What is your favorite topic?\r"
send -- "$my_favorite\r"
expect eof
```

Now try to run the Expect script with some parameters to see the output:

```
$ ./answerbot SomeName Programming
```

A screenshot of a terminal window titled 'likegeeks@likegeeks-VirtualBox ~/Desktop'. The window has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal shows a command prompt where the user runs './answerbot SomeName Programming'. The script responds with a series of questions and answers: 'spawn ./questions', 'Hello, who are you?', 'Im SomeName', 'Can I ask you some questions?', 'Sure', 'What is your favorite topic?', and 'Programming'. The prompt returns to the user.

```
likegeeks@likegeeks-VirtualBox ~/Desktop $ ./answerbot SomeName Programming
spawn ./questions
Hello, who are you?
Im SomeName
Can I ask you some questions?
Sure
What is your favorite topic?
Programming
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Awesome!! Now our automated Expect script is more dynamic.

Advertisements

Conditional Tests

You can write conditional tests using braces like this:

```
expect {
    "something" { send -- "send this\r" }
    "*another" { send -- "send another\r" }
}
```

We are going to change our script to return different conditions, and we will change our Expect script to handle those conditions.

We are going to emulate different expects with the following script:

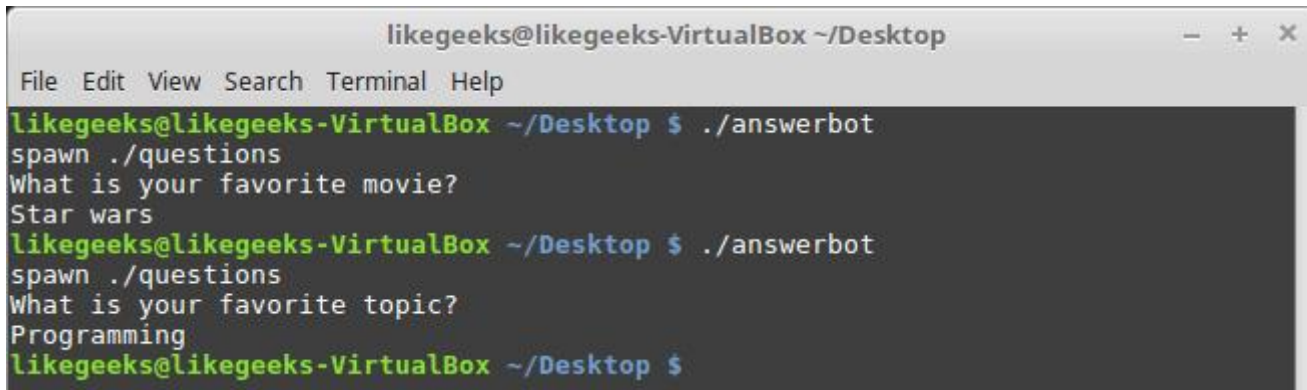
```
#!/bin/bash
let number=$RANDOM
if [ $number -gt 25000 ]
then
echo "What is your favorite topic?"
else
echo "What is your favorite movie?"
fi
read $REPLY
```

A random number is generated every time you run the script and based on that number, we put a condition to return different expects.

Let's make out Expect script that will deal with that.

```
#!/usr/bin/expect -f
set timeout -1
spawn ./questions
expect {
    "*topic?" { send -- "Programming\r" }
    "*movie?" { send -- "Star wars\r" }
```

```
}  
expect eof
```



```
likegeeks@likegeeks-VirtualBox ~/Desktop  
File Edit View Search Terminal Help  
likegeeks@likegeeks-VirtualBox ~/Desktop $ ./answerbot  
spawn ./questions  
What is your favorite movie?  
Star wars  
likegeeks@likegeeks-VirtualBox ~/Desktop $ ./answerbot  
spawn ./questions  
What is your favorite topic?  
Programming  
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Very clear. If the script hits the topic output, the Expect script will send programming and if the script hits movie output the expect script will send star wars. Isn't cool?

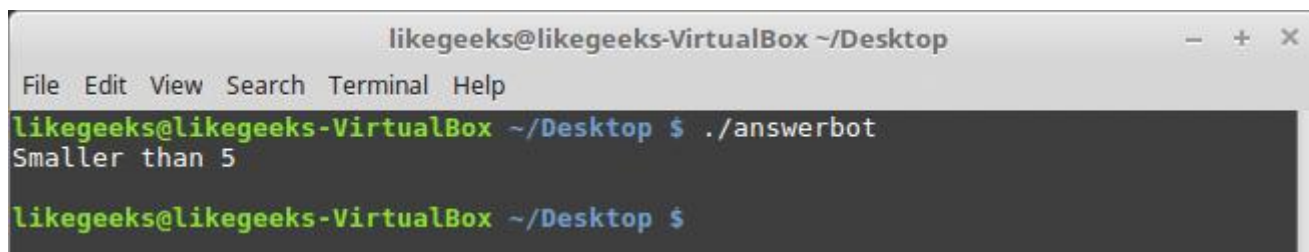
Advertisements

If else Conditions

You can use if/else clauses in expect scripts like this:

```
#!/usr/bin/expect -f  
  
set NUM 1  
  
if { $NUM < 5 } {
```

```
puts "\Smaller than 5\n"  
} elseif { $NUM > 5 } {  
puts "\Bigger than 5\n"  
} else {  
puts "\Equals 5\n"  
}
```

A screenshot of a terminal window titled 'likegeeks@likegeeks-VirtualBox ~/Desktop'. The window has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal shows the command './answerbot' being executed, which outputs 'Smaller than 5'. The prompt returns to the shell.

```
likegeeks@likegeeks-VirtualBox ~/Desktop  
File Edit View Search Terminal Help  
likegeeks@likegeeks-VirtualBox ~/Desktop $ ./answerbot  
Smaller than 5  
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Note: **The opening brace must be on the same line.**

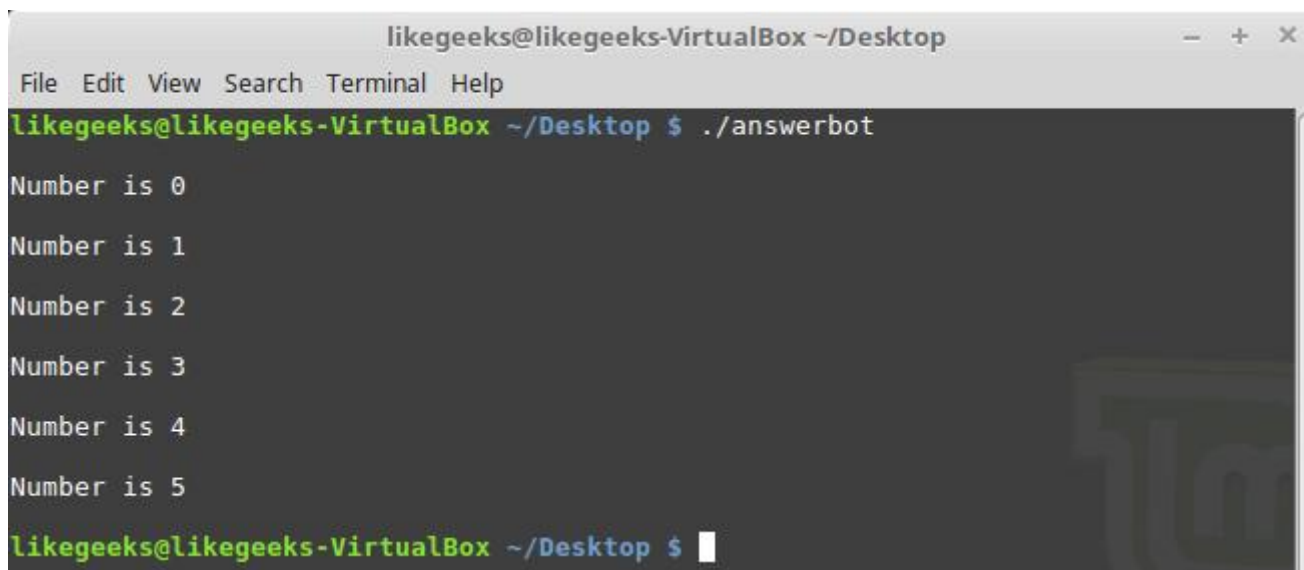
Advertisements

While Loops

While loops in expect language must use braces to contain the expression like this:

```
#!/usr/bin/expect -f  
  
set NUM 0
```

```
while { $NUM <= 5 } {  
  puts "\nNumber is $NUM"  
  set NUM [ expr $NUM + 1 ]  
}  
puts ""
```

A screenshot of a terminal window titled 'likegeeks@likegeeks-VirtualBox ~/Desktop'. The window has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal shows the command './answerbot' being executed, which results in six lines of output: 'Number is 0', 'Number is 1', 'Number is 2', 'Number is 3', 'Number is 4', and 'Number is 5'. The prompt 'likegeeks@likegeeks-VirtualBox ~/Desktop \$' is visible at the bottom.

```
likegeeks@likegeeks-VirtualBox ~/Desktop  
File Edit View Search Terminal Help  
likegeeks@likegeeks-VirtualBox ~/Desktop $ ./answerbot  
Number is 0  
Number is 1  
Number is 2  
Number is 3  
Number is 4  
Number is 5  
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Advertisements

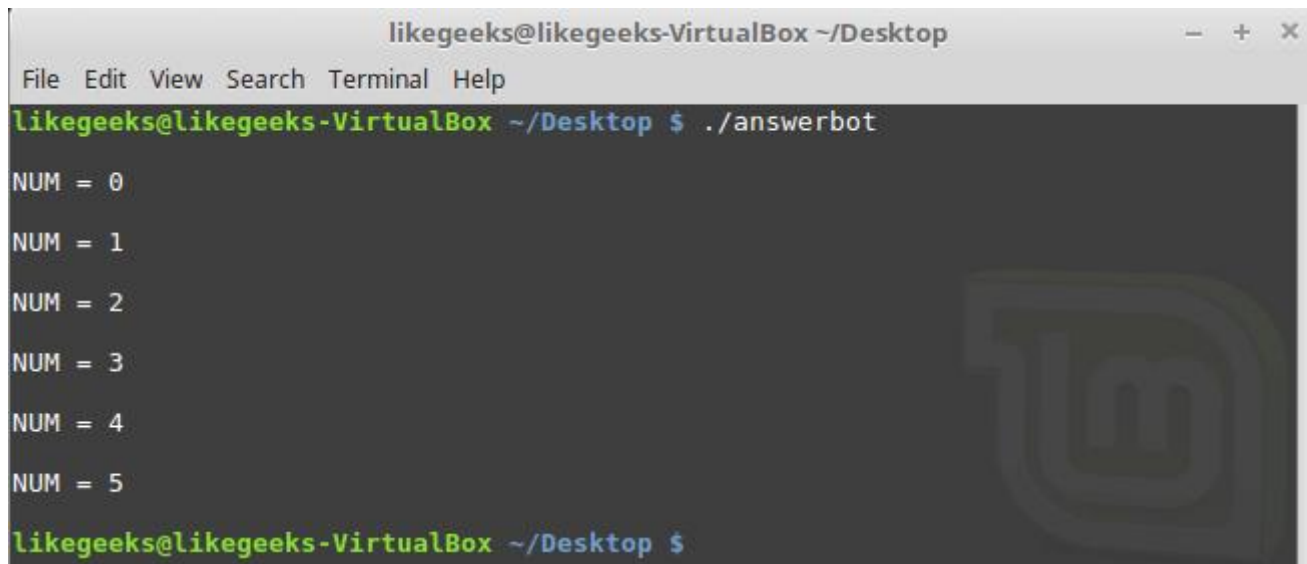
For Loops

To make a for loop in expect, three fields must be specified, like the following format:

```
#!/usr/bin/expect -f

for {set NUM 0} {$NUM <= 5} {incr NUM} {
puts "\nNUM = $NUM"
}

puts ""
```

A screenshot of a terminal window titled 'likegeeks@likegeeks-VirtualBox ~/Desktop'. The window has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal shows the command './answerbot' being executed. The output is a loop printing 'NUM = 0' through 'NUM = 5'. A large, faint watermark of a Twitter bird logo is visible in the background of the terminal output.

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ ./answerbot
NUM = 0
NUM = 1
NUM = 2
NUM = 3
NUM = 4
NUM = 5
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

User-defined Functions

You can define a function using proc like this:

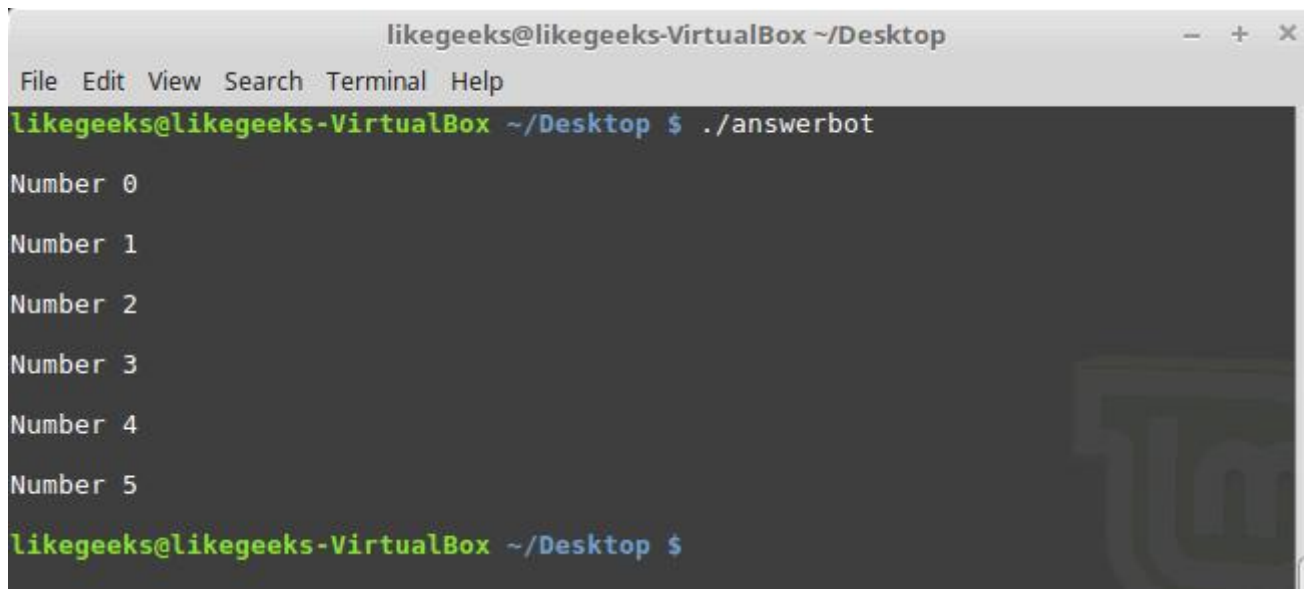
```
proc myfunc { TOTAL } {
set TOTAL [expr $TOTAL + 1]
```



```
return "$TOTAL"  
}
```

And you can use them after that.

```
#!/usr/bin/expect -f  
proc myfunc { TOTAL } {  
    set TOTAL [expr $TOTAL + 1]  
    return "$TOTAL"  
}  
set NUM 0  
while {$NUM <= 5} {  
    puts "\nNumber $NUM"  
    set NUM [myfunc $NUM]  
}  
puts ""
```



```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ ./answerbot
Number 0
Number 1
Number 2
Number 3
Number 4
Number 5
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Interact Command

Sometimes your Expect script contains some sensitive information that you don't want to share with other users who use your Expect scripts, like passwords or any other data, so you want your script to take this password from you and continuing automation normally.

The interact command reverts the control back to the keyboard.

When this command is executed, Expect will start reading from the keyboard.

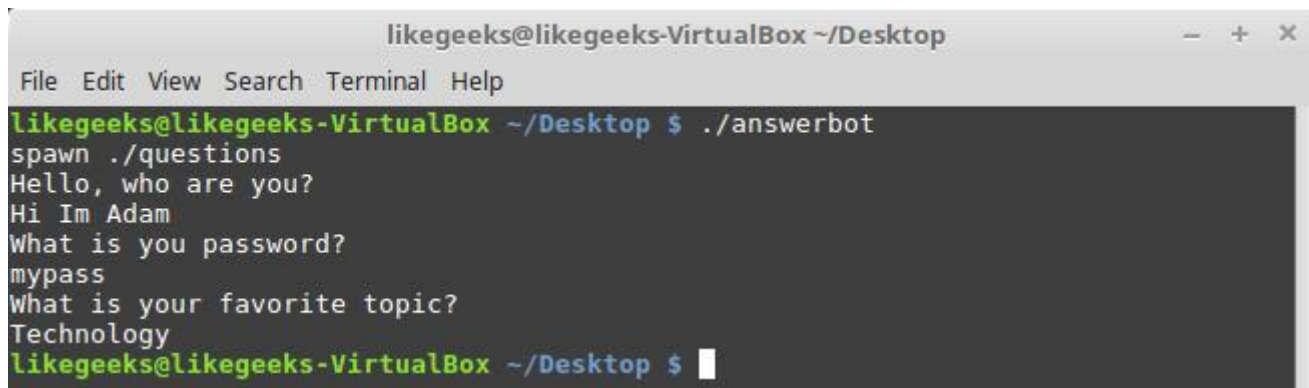
This shell script will ask about the password as shown:

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

```
echo "Hello, who are you?"  
read $REPLY  
echo "What is your password?"  
read $REPLY  
echo "What is your favorite topic?"  
read $REPLY
```

Now we will write the Expect script that will prompt for the password:

```
#!/usr/bin/expect -f  
set timeout -1  
spawn ./questions  
expect "Hello, who are you?\r"  
send -- "Hi Im Adam\r"  
expect "*password?\r"  
interact ++ return  
send "\r"  
expect "*topic?\r"  
send -- "Technology\r"  
expect eof
```

A screenshot of a terminal window titled 'likegeeks@likegeeks-VirtualBox ~/Desktop'. The window has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal shows the following interaction:

```
likegeeks@likegeeks-VirtualBox ~/Desktop $ ./answerbot
spawn ./questions
Hello, who are you?
Hi Im Adam
What is you password?
mypass
What is your favorite topic?
Technology
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

After you type your password type ++ and the control will return back from the keyboard to the script.

Expect language is ported to many languages like C#, Java, Perl, **Python**, Ruby and Shell with almost the same concepts and syntax due to its simplicity and importance.

Expect scripting language is used in quality assurance, network measurements such as echo response time, automate file transfers, updates, and many other uses.

I hope you now supercharged with some of the most important aspects of Expect command, autoexpect command and how to use it to automate your tasks in a smarter way.

Thank you.

**admin**<https://likegeeks.com>

Related Articles



Linux

Learn Linux Environment Variables Step-By-Step Easy Guide



 2017-02-04  admin

In the previous posts, we talked about some of the basic Linux commands, today we continue our journey, and we will talk about something very important in Linux which is Linux



Linux

How to write practical shell scripts



 2017-02-25  admin

In the last post, we talked about regular expressions and we saw how to use them in sed and awk for text processing, and we discussed before Linux sed command and awk command. During the series, we wrote



Linux

What is Linux File System? Easy Guide

 2017-01-30  admin

As we've talked about Linux on the previous post and we have chosen the best Linux distro, also we've learned how to install Linux. Today we will discuss the Linux file system.

Advertisements What Is Linux File

Environment Variables. So what are Environment Variables and what is the benefit of knowing them?

Advertisements Environment Variables are used to store some values [...]

◀ 61

[◀ How to write practical shell scr...](#)

small shell scripts, but we didn't mix things up, I think we should take a small step further [...]

◀ 213

System? Linux File System or any file system generally is a layer which is under the operating system that [...]

◀ 67

[Performance Tuning Using Linux ...](#)

0 Comments **likegeeks****1 Login** ▼♥ Recommend 2  Share

Sort by Best ▼



Start the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS 

Be the first to comment.

ALSO ON LIKEGEEKS

Define and Use Tensors Using Simple TensorFlow Examples

4 comments • 10 months ago



Lee Slash — Thank you for your reply.I have try this way, but there are still some questions,1.This step just read data into structure, why should we us coord = ...

Secure Linux Server Using Hardening Best Practices

2 comments • a year ago



Jim Dennis — That find command is way too complicated. This will suffice: find / -type f -perm +6000 -ls... the + prefix on the permissions (octal) causes find ...

Install and Configure Linux LDAP Server

6 comments • a year ago



Delal Khir — great thank's

Useful Linux Security Tricks to Harden Your System

16 comments • a year ago



likegeeks — What a valuable lesson! Thanks to VMsAnyway, I'm glad that you learned a new thing.Regards,

 **Subscribe**  **Add Disqus to your site**Add DisqusAdd  **Disqus' Privacy Policy**Privacy PolicyPrivacy PolicyPrivacy Policy

Subscribe to My Newsletter

Email *

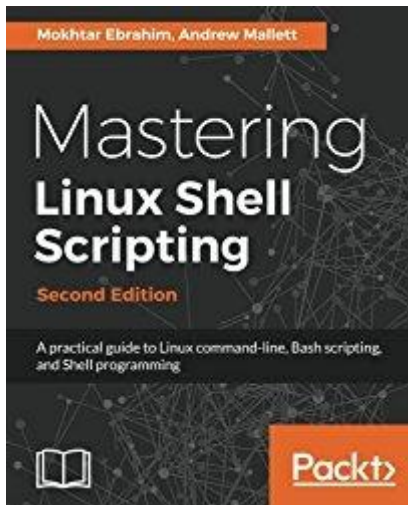
Subscribe

Advertisements

Donate via PayPal

Donate

My Published Book



Latest Posts



Python GUI examples (Tkinter Tutorial)

📅 2018-01-22 👤 admin

20+ Python Web Scraping Examples (Beautiful Soup & Selenium)

📅 2017-12-05 👤 admin



NLP Tutorial Using Python NLTK (Simple Examples)

📅 2017-09-21 👤 admin

Define and Use Tensors Using Simple TensorFlow Examples

📅 2017-08-16 👤 admin

Cast or Convert an Array to Object Using PHP (Hydrator Pattern)



Expect command and how to automate shell scripts like magic - Like Geeks

📅 2017-05-27 👤 admin

Advertisements

See Also



How to write practical shell scripts

📅 2017-02-25 👤 admin

Regex tutorial for Linux (Sed & AWK) examples



Expect command and how to automate shell scripts like magic - Like Geeks

📅 2017-02-23 👤 admin



30 Examples for Awk Command in Text Processing

📅 2017-02-21 👤 admin



31+ Examples for sed Linux Command in Text Manipulation

📅 2017-02-19 👤 admin

Bash Scripting Part6 – Create and Use Bash Functions



📅 2017-02-17 👤 admin