

## Virtual Machine

A virtual machine (VM) is a computer program that simulates a computer. The VM software we're using in this course is called **VirtualBox**. When you set up your virtual machine you installed Linux on the VM, making Linux the **guest operating system**. The operating system (OS) that's installed directly on your physical computer is called the **host OS**.

We use a virtual machine in this course to ensure that everyone is working in an identical environment with the correct programs installed, but there are many other reasons programmers use VMs.

**VMs isolate programming projects** from everything else on a programmer's computer. The programmer can configure the guest OS by installing programs and customizing settings without disrupting their day-to-day environment.

**VMs are also used to simulate the environment that software will be deployed to.** Most developers use Windows or Mac OS, but often deploy their code to servers running Linux. Using a Linux VM lets programmers run code on their target platform, without leaving the comfort of their preferred host OS.

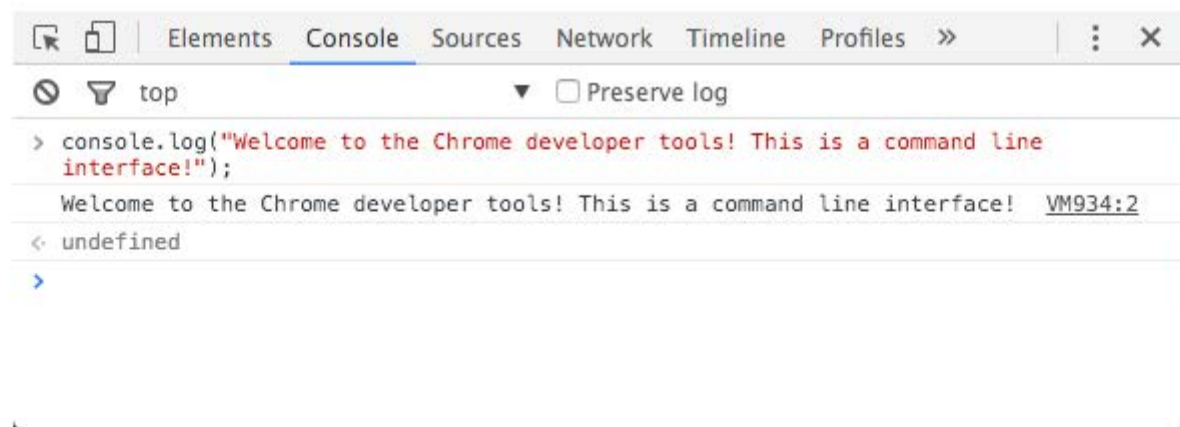
## Vagrant

Vagrant is a program that makes VMs more convenient to use. For example when you ran `vagrant up` Vagrant created a VM, installed a guest OS, and configured the guest OS. Vagrant did all of this automatically by following instructions in the Vagrantfile. Automating this process saves time and ensures consistent results.

Vagrant also makes it easy to edit files that are in the VM from programs installed on the host OS. We won't use this feature in this class, but it's very helpful in other Udacity courses and on the job.

## Command Line Interface

Programmers encounter many different command line interfaces (**CLIs**) in their work. Any computer interface where the user enters textual commands and gets textual responses is a CLI. While CLIs vary significantly, proficiency in one will give you a head start learning another. In this class we mostly work with the Linux command line interface in a VM, but in order to access that you need to use your host OS's command line interface. Other command line interfaces you might encounter as a developer are your browser's tools and Python's interactive interpreter.



A screenshot of Chrome Developer Tools' console, which is a sort of command line interface with your browser.

```
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 5 2015, 21:12:44)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Welcome to the Python interpreter!")
Welcome to the Python interpreter!
>>> print("Yet another interactive CLI")
Yet another interactive CLI
>>> □
```

Different programming languages will have different Command Line Interfaces (CLIs). A screenshot of an interactive Python session on the command line using the Python CLI. Instead of using a software program with a more standard visual user interface, many programmers often use the command line with a CLI as shown above.



## Your own Linux box

To learn the Linux shell, you need a Linux machine to run it on. But we can't really ship a new Linux computer to every one of you. So instead you will set up a Linux *virtual machine* (VM) on your own computer.

You'll be using the [VirtualBox](#) application to run the virtual machine, and the [vagrant](#) software to configure it.

This virtual-machine setup is very similar to the ones you will use in later Udacity courses on the Linux platform. So when you get to those courses, you will not need to re-install this software.

Setting the virtual machine up is not complicated, but it will take some time when your computer downloads the Linux OS. Follow the instructions below to set it up before proceeding on in this course.

## What's a virtual machine?

A virtual machine is a program that runs on your Windows or Mac computer, and that can run a different operating system inside it. In this case, you'll be running an Ubuntu Linux server system.

## 1. Install Git

*You can skip this step if you are not running Windows, but many other courses use Git, so you may want to install it now.*

Download Git from [git-scm.com](https://git-scm.com). Install the version for your operating system.

On Windows, Git will provide you with the **Git Bash** terminal program, which you will use to run and connect to your Linux VM.

## 2. Find your terminal program

To take this course you will need to use a **terminal program**, which presents the shell user interface and lets you log in to your Linux VM.

- Windows: Use the **Git Bash** program, which is installed with Git (above).
- Mac OS X: Use the **Terminal** program, located in your **Applications/Utilities** folder.
- Linux: Use any terminal program (e.g. **xterm** or **gnome-terminal**).

## 3. Install VirtualBox

VirtualBox is the software that actually runs the VM. [You can download it from virtualbox.org, here.](https://www.virtualbox.org)

Install the *platform package* for your operating system. You do not need the extension pack or the SDK. You do not need to launch VirtualBox after installing it.

**Ubuntu 14.04 Note:** If you are running Ubuntu 14.04, install VirtualBox using the Ubuntu Software Center, not the virtualbox.org web site. Due to a [reported bug](#), installing VirtualBox from the site may uninstall other software you need.

## 4. Install Vagrant

Vagrant is the software that configures the VM and lets you share files between your host computer and the VM's filesystem. [You can download it from vagrantup.com](#). Install the version for your operating system.

**Windows Note:** The Installer may ask you to grant network permissions to Vagrant or make a firewall exception. Be sure to allow this.

## 5. Download the VM configuration file

Make a new folder to keep your workspace for this course. You might call it **Shell**, but whatever name you pick is OK. Keep track of what folder you created it in (for instance, **Desktop**).

The configuration file, called `Vagrantfile`, is here: [Download!](#)

Download this file into the new folder you just created.



## 6. Run the virtual machine!

Open your terminal program. Type this shell command and press Enter:

```
cd Desktop/Shell
```

(If your new folder is called something other than "Shell", or is located somewhere other than "Desktop", change those.)

Then start the VM by running the command `vagrant up`.

This will make your system download the Linux OS and start up the virtual machine. Unfortunately, this will take a long time on most network connections. Fortunately, you only have to do it once, and the same Linux OS image will work for later Udacity courses too.

Once it is done, run the command `vagrant ssh`.

And you will be logged in to the virtual machine and ready to do the course exercises!

The Udacity VM is the official shell for this class, but if your computer already has a Unix\* shell you can use it if you prefer.

**Caveat:** Your computer's own shell may differ from the VM in unanticipated ways, and may not have all the programs installed which the VM provides. The recommended environment is the VM.

\* if you're running Linux or Mac OS X for instance





## Log In and Break Stuff

Type things into the shell until you find something that "breaks" it — something that makes the shell prompt not come back when you press Enter.

What was it that you did?

```
vagrant@vagrant-ubuntu-trusty-64:~$ Alice's tea party  
> █
```

That'll happen if you type something  
with a single quote mark, or



0:07 / 0:56



YouTube



```
vagrant@vagrant-ubuntu-trusty-64:~$ Alice's tea party
```

```
> █
```

with a parenthesis, or brace.



0:10 / 0:56



YouTube



```
vagrant@vagrant-ubuntu-trusty-64:~$ Alice's tea party
```

```
> '
```

you have to type the matching quote  
mark, or parenthesis, or brace.



0:16 / 0:56



YouTube





```
vagrant@vagrant-ubuntu-trusty-64:~$ Alice's tea party
>
Alices tea party
: command not found
vagrant@vagrant-ubuntu-trusty-64:~$ exit
logout
Connection to 127.0.0.1 closed.
vm$ vagrant ssh
```

Now, if you typed exit, or  
if you type Ctrl+D, you logged out.



0:26 / 0:56



YouTube



22 updates are security updates.

## Log In and Break Stuff



Watch later

Share

Last login: Tue Jul 7 21:47:56 2015 from 10.0.2.2

vagrant@vagrant-ubuntu-trusty-64:~\$ bc

bc 1.06.95

Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006 Free Software Foundation, Inc.

This is free software with ABSOLUTELY NO WARRANTY.

For details type `warranty'.

kjsdfkjhsdkjfh

0

iuhfrihf

0

1 + 1

2

+++

(standard\_in) 4: syntax error

klfkjg

0

█

going on,

you can usually type Ctrl+C or look for



0:35 / 0:56



YouTube





Watch later

Share

Log In and Break Stuff

vagrant@vagrant-ubuntu-trusty-64:~\$ bc

bc 1.06.95

Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006 Free Software Foundation, Inc.

This is free software with ABSOLUTELY NO WARRANTY.

For details type `warranty'.

kjsdfkjhsdkjfh

0

iuhfrihf

0

1 + 1

2

+++

(standard\_in) 4: syntax error

klfkjg

0

^C

(interrupt) use quit to exit.

quit



0:40 / 0:56



YouTube

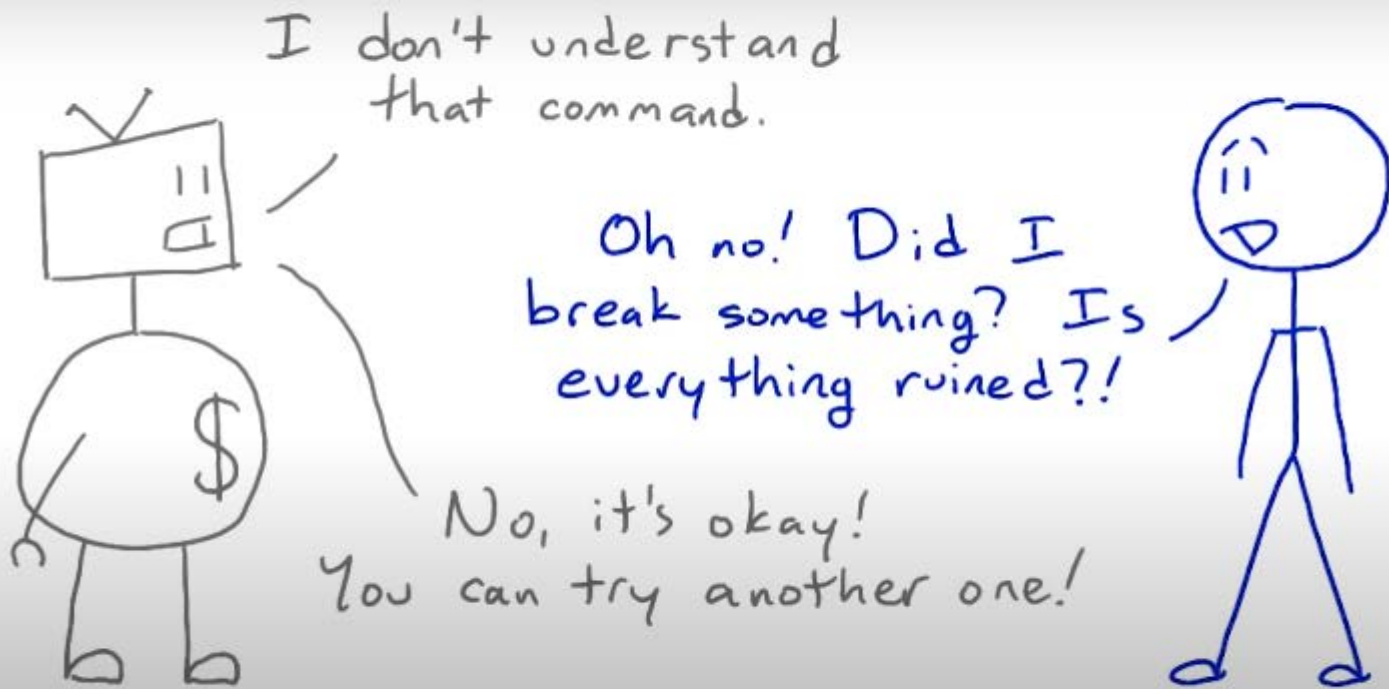




# Log In and Break Stuff



Watch later Share



0:56 / 0:56



YouTube





## In the VM or out of the VM?

We've set this course's exercises up to work in the virtual machine (VM) that you set up using the **vagrant** program. If you get logged out of the VM, you may end up typing shell commands in to your regular operating system instead of to the Linux system that we've set up for the course. Some commands won't work, and some files probably won't be where the course expects them to be.

### Getting logged out

If you type the command **exit** into the shell, or if you type **Control-D**, you will see a message like this:

```
logout
Connection to 127.0.0.1 closed.
```

This just means that you got logged out. After logging out, you won't be in the VM any more.

To get back into the VM, use the command `vagrant ssh`.

### If `vagrant ssh` doesn't work

If you get a message like this:

```
VM must be running to open SSH connection. Run `vagrant up`
to start the virtual machine.
```

This means that the VM program is not running, for instance because you rebooted your computer. This is just fine and it doesn't mean you've lost any work. Just run `vagrant up` to bring the VM back up, then `vagrant ssh` to log in.

This will not take as long as the first time you ran it, because it won't need to download the Linux OS.

### If `vagrant up` doesn't work

If you get a message like this:

```
A Vagrant environment or target machine is required to run this
command. Run `vagrant init` to create a new Vagrant environment. Or,
get an ID of a target machine from `vagrant global-status` to run
this command on. A final option is to change to a directory with a
Vagrantfile and to try again.
```

That means that **vagrant** can't find the configuration file you downloaded. Go back to [the instructions](#), check to be sure that you did step 5, and then do step 6 again.

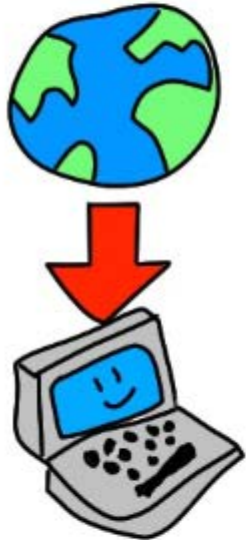
### Multiple terminals

If you open up more than one terminal window, only the one(s) that you ran `vagrant ssh` in will be connected to your Linux OS for this course. The others will be connected to your regular OS.

(It's actually really normal for Linux users to have to carefully keep track of which terminal windows are connected to which machines. Don't panic. Just look for whether "vagrant" appears on the command line.)

What do you think this command will do?

```
vagrant@vagrant-ubuntu-trusty-64:~$ ls  
vagrant@vagrant-ubuntu-trusty-64:~$ curl http://udacity.github.io/ud595-shell/stuff.zip  
-o things.zip
```



- ☐ Send all your money to Karl so he can buy more silly T-shirts
- ☐ Curl your hair, put stuff in it, and snag it in a zipper
- ☐ Download a file from the web



[View Intro](#)

[VIEW ANSWER](#)

[SUBMIT ANSWER](#)

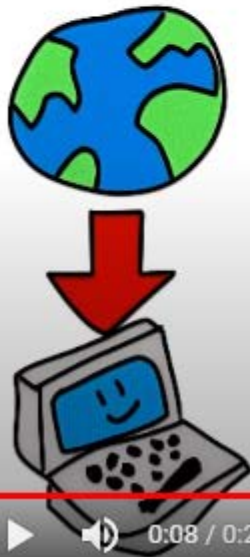
Here is the command that Philip asks you to run:

```
curl http://udacity.github.io/ud595-shell/stuff.zip -o things.zip
```



What do you think this command will do?

```
vagrant@vagrant-ubuntu-trusty-64:~$ ls  
things.zip  
vagrant@vagrant-ubuntu-trusty-64:~$ curl http://udacity.github.io/ud595-shell/stuff.zip  
-o things.zip
```



- ☐ Send all your money to Karl so he can buy more silly T-shirts
- ☐ Curl your hair, put stuff in it, and snap it in a zipper
- ☒ Download files from the web

Curl is a command for

downloading files from the web



0:08 / 0:20

CC BY-NC-SA YouTube



# What can you do in the terminal?



Graphical Interface

```
user@host:~$ ls
thing  stuff  program
user@host:~$ ./program
Hello, world!
user@host:~$ █
```

Terminal

interfaces are the most common  
interfaces for desktop programs.

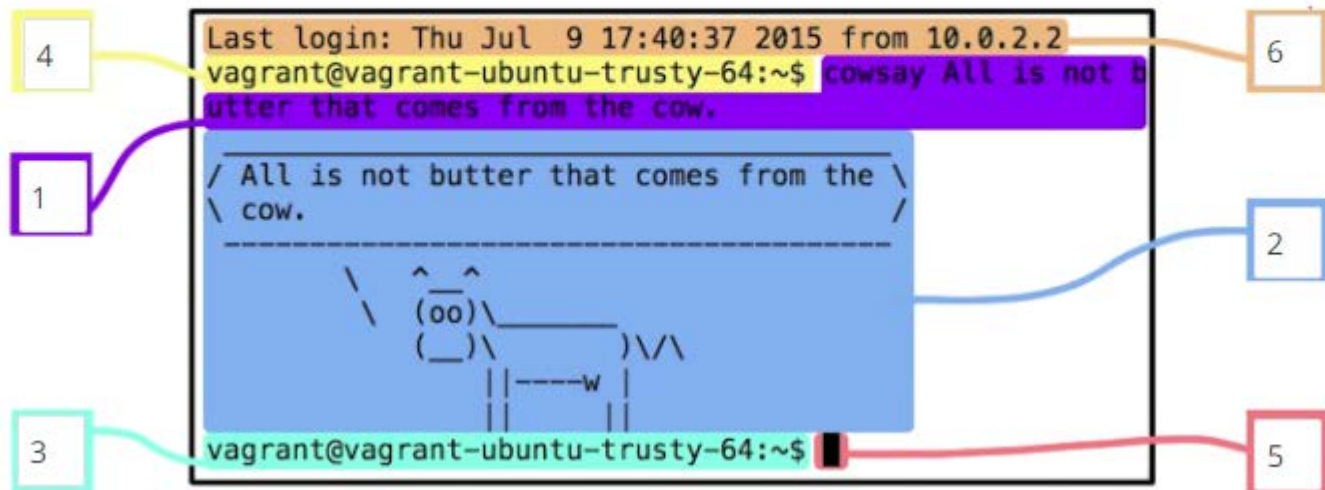
# What can you do in the terminal?

Which of these activities do you expect can be readily done using the terminal?

- ☒ Edit and run a program.
- ☒ Find files on your computer that have particular names.
- ☒ Download a file from the web, if you know its URL.
- ☐ Edit a major motion picture.
- ☒ Start a web server on your computer.
- ☒ Start a web server in the cloud.
- ☐ Break the entire Internet.

# Label the interface elements in this terminal:

- 1: shell command      3: current prompt      5: cursor  
2: command output    4: previous prompt    6: login message



Enter the number of the corresponding interface element in each box

If you'd like to use the cowsay program outside of the VM, on your own computer, you can install it like this:

**Ubuntu and Debian users:** `sudo apt-get install cowsay`

**Redhat and CentOS users:** `sudo yum install cowsay`

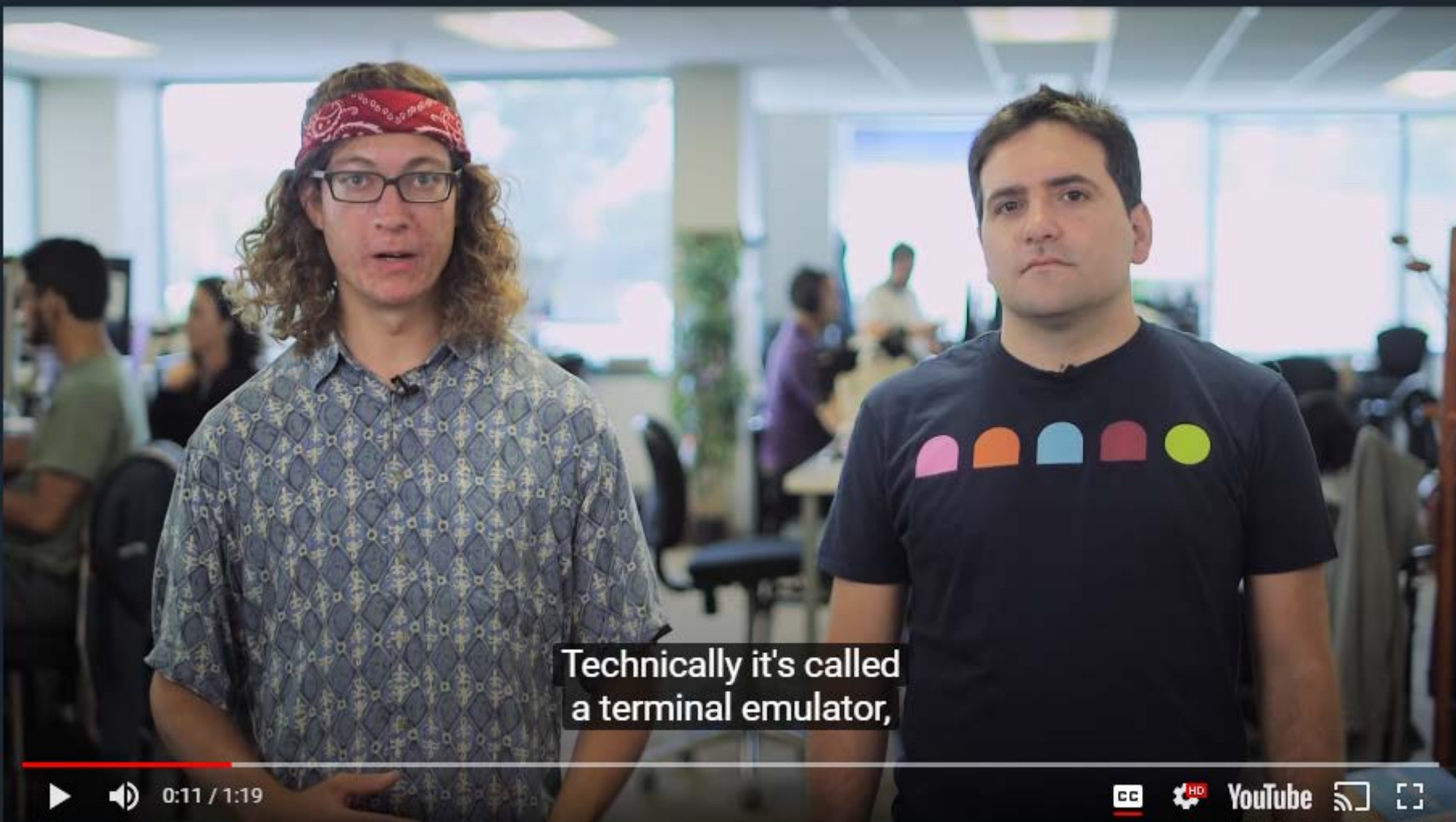
**OS X users:** `brew install cowsay`

(This requires the homebrew, a third party package manager for OS X, <http://brew.sh/>)

**Arch Linux users:** `sudo pacman -S cowsay`

Note: You typically need to be a “superuser” to install new software, that’s why these install commands begin with the sudo command . You can learn more about sudo in our [Configuring Linux Web Servers](#) course or on [Wikipedia](#).






0:11 / 1:19



YouTube







The default one on most Linux systems,  
and on the Mac, is called GNU Bash.



0:57 / 1:19



YouTube



But there are others called TCSH,  
KSH and Seashell.



0:58 / 1:19



YouTube



This is called shell scripting.



1:15 / 1:19



YouTube





## Different shells

Unix and Linux programmers over the years have written many different shell programs. You can read more about them on Wikipedia: the original [Bourne shell](#) or **sh**; the [C shell](#) or **cs****h**; the [Korn shell](#) or **ks****h**; the [Z shell](#) or **zs****h**; as well as the **ba****sh** shell that this course uses.

Different systems may have different shells installed by default. Most Linux systems, and Mac OS X, default to **ba****sh** for interactive shells. However, the most common default shell for scripting (shell programming) is classic **sh**. BSD Unix systems usually default to **sh** or **ks****h**.

Almost everything in this course will work the same in any of these shell programs. The exception is one of the file matching (globbing) syntaxes at the end of Lesson 3.



# Try More Commands!

date

hostname

expr 2 + 2

host udacity.com

echo You rock

bash --version

uname

history

These are called Command Line Arguments.



0:24 / 0:46



YouTube



vagrant@vagrant-ubuntu-trusty-64:~\$ expr 2 + 2

4

vagrant@vagrant-ubuntu-trusty-64:~\$ echo You rock

You rock

vagrant@vagrant-ubuntu-trusty-64:~\$ echo You rock

You rock

vagrant@vagrant-ubuntu-trusty-64:~\$ echo You rock

You rock

vagrant@vagrant-ubuntu-trusty-64:~\$ echo You rock

You rock

vagrant@vagrant-ubuntu-trusty-64:~\$ **uname**

Linux

vagrant@vagrant-ubuntu-trusty-64:~\$ uname -a

Linux vagrant-ubuntu-trusty-64 3.13.0-55-generic #92-Ubuntu SMP Sun Jun 14 18:32:20 UTC

2015 x86\_64 x86\_64 x86\_64 GNU/Linux

vagrant@vagrant-ubuntu-trusty-64:~\$ █



0:38 / 1:41



YouTube





```
vagrant@vagrant-ubuntu-trusty-64:~$ uname -a
Linux vagrant-ubuntu-trusty-64 3.13.0-55-generic #92-Ubuntu SMP Sun Jun 14 18:32:20 UTC
2015 x86_64 x86_64 x86_64 GNU/Linux
vagrant@vagrant-ubuntu-trusty-64:~$ hostname
vagrant-ubuntu-trusty-64
vagrant@vagrant-ubuntu-trusty-64:~$ host udacity.com
udacity.com has address 50.116.54.191
udacity.com mail is handled by 20 ALT2.ASPMX.L.GOOGLE.com.
udacity.com mail is handled by 30 ASPMX2.GOOGLEMAIL.com.
udacity.com mail is handled by 30 ASPMX3.GOOGLEMAIL.com.
udacity.com mail is handled by 10 ASPMX.L.GOOGLE.com.
udacity.com mail is handled by 20 ALT1.ASPMX.L.GOOGLE.com.
vagrant@vagrant-ubuntu-trusty-64:~$ bash --version
GNU bash, version 4.3.11(1)-release (x86_64-pc-linux-gnu)
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
```

This is free software; you are free to change and redistribute it.  
There is NO WARRANTY, to the extent permitted by law.

vagrant@vagrant-ubuntu-trusty-64:~\$ **often with an option like `--version` or  
dash capital V.**



1:29 / 1:41



YouTube



```
2015 x86_64 x86_64 x86_64 GNU/Linux
vagrant@vagrant-ubuntu-trusty-64:~$ hostname
vagrant-ubuntu-trusty-64
vagrant@vagrant-ubuntu-trusty-64:~$ host udacity.com
udacity.com has address 50.116.54.191
udacity.com mail is handled by 20 ALT2.ASPMX.L.GOOGLE.com.
udacity.com mail is handled by 30 ASPMX2.GOOGLEMAIL.com.
udacity.com mail is handled by 30 ASPMX3.GOOGLEMAIL.com.
udacity.com mail is handled by 10 ASPMX.L.GOOGLE.com.
udacity.com mail is handled by 20 ALT1.ASPMX.L.GOOGLE.com.
vagrant@vagrant-ubuntu-trusty-64:~$ bash --version
GNU bash, version 4.3.11(1)-release (x86_64-pc-linux-gnu)
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
```

This is free software; you are free to change and redistribute it.

There is NO WARRANTY, to the extent permitted by law.

```
vagrant@vagrant-ubuntu-trusty-64:~$ python -V
```

```
Python 2.7.6
```

```
vagrant@vagrant-ubuntu-trusty-64:~$
```

Python -V tells us Python 2.7.6 and



1:35 / 1:41



YouTube



## Try More Commands!

Try each of these commands, and describe what it does.

date

prints date and time

expr 2 + 2

add 2 and 2, print 4

echo You rock

prints "You rock"

uname

prints "Linux" (the OS name)

hostname

prints the Vagrant VM's name

host udacity.com

gives Udacity's IP address

bash --version

prints bash version & copyright

history

the history command tells you all the commands you've run previously.



which stands for remove And  
it works like this.

```
$ rm weirdfile.txt
```



0:19 / 0:36



YouTube



>> And you could do the same thing in  
Python with the remove function in

```
>>> os.remove("weirdfile.txt")
```



0:23 / 0:36



YouTube





>> Like Carl says, running a Shell  
command in your terminal is a lot like



0:19 / 0:38



YouTube





calling a function in a program.

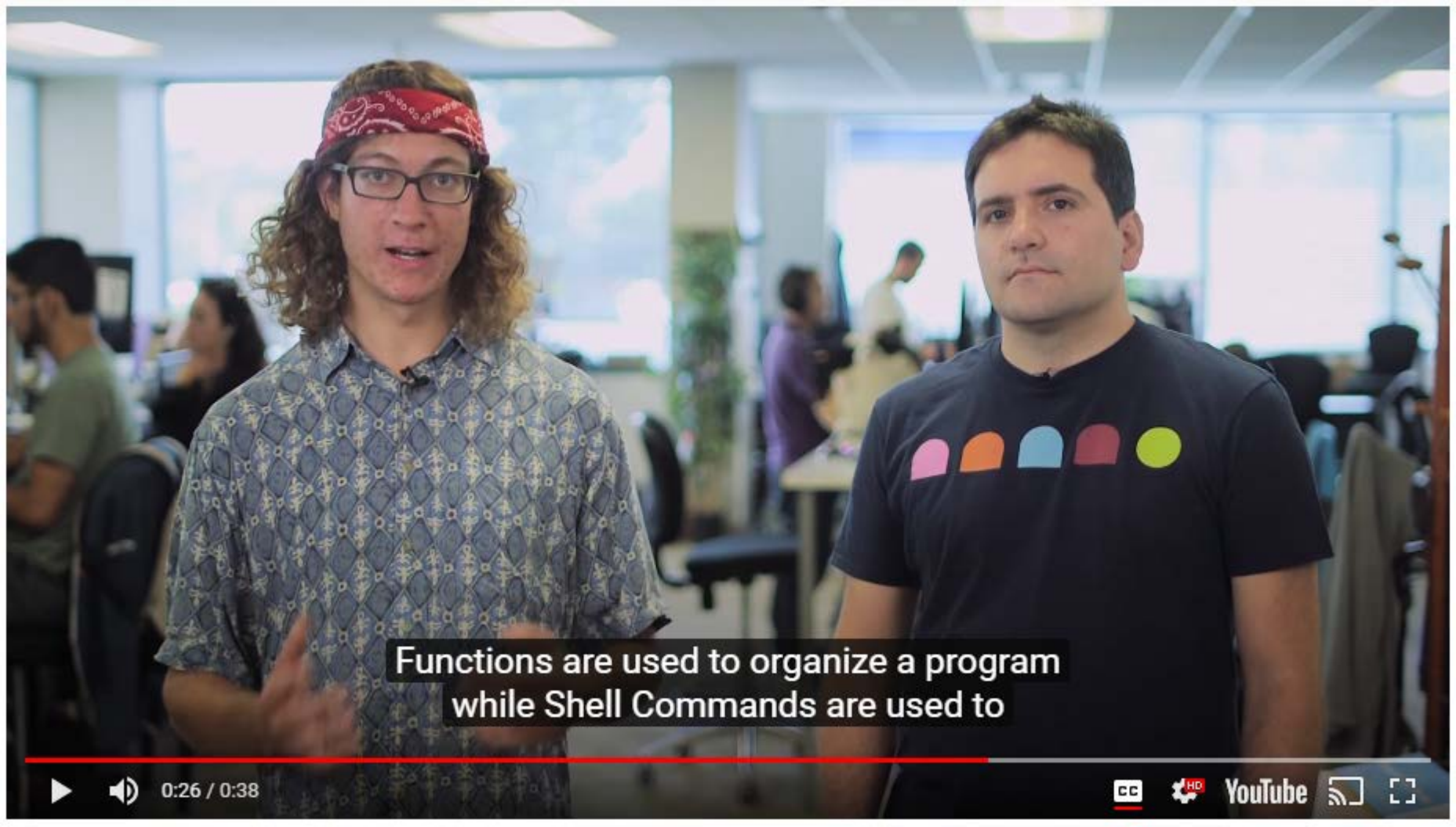


0:22 / 0:38



YouTube





Functions are used to organize a program  
while Shell Commands are used to



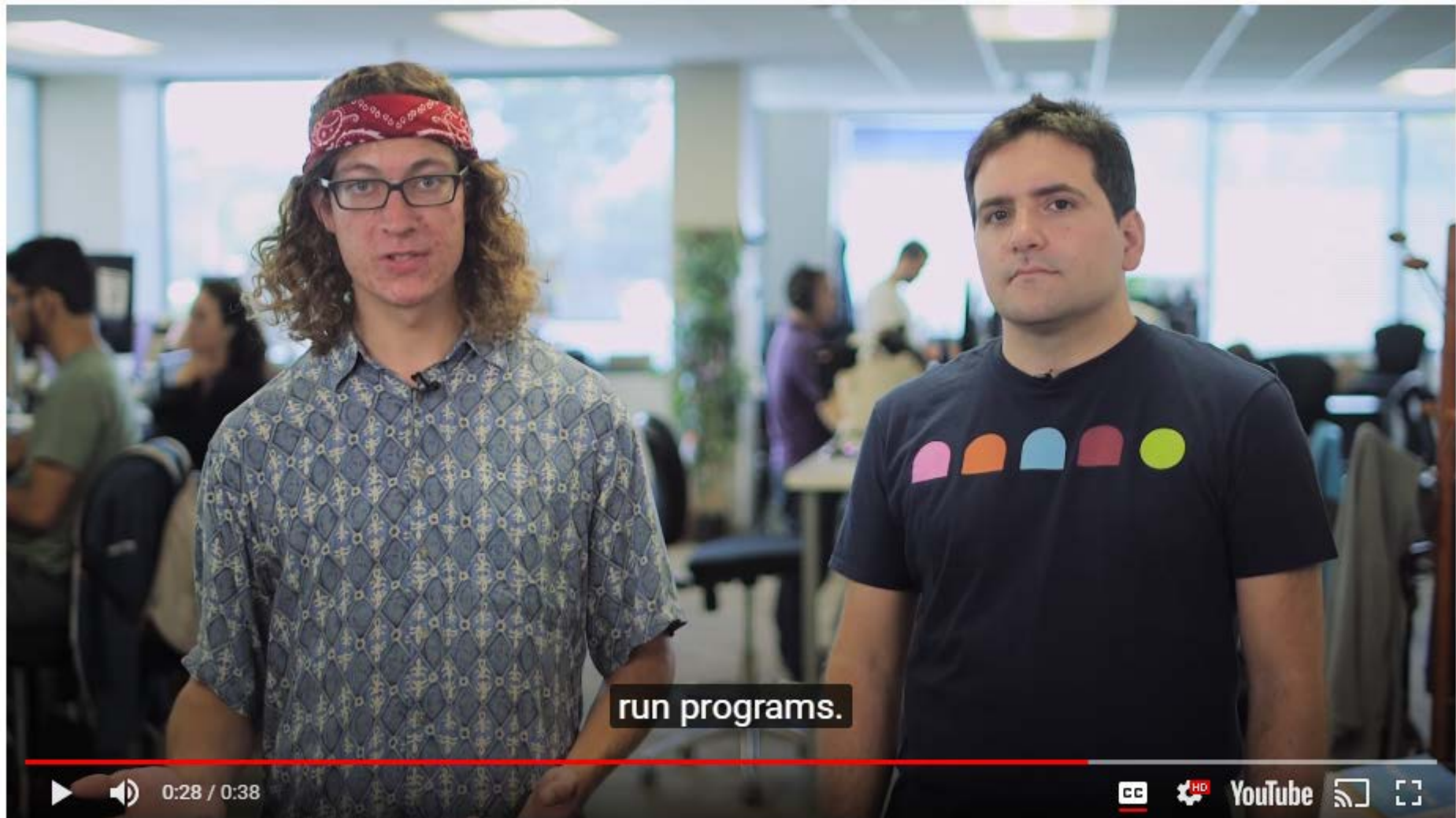
0:26 / 0:38



YouTube







run programs.



0:28 / 0:38



YouTube



# What is a shell command?

Describe what a shell command is.

How is it similar to, and different from, a function in a language such as Python or JavaScript?





1. vagrant@vagrant-ubuntu-trusty-64: ~ (ssh)

```
vagrant@vagrant-ubuntu-trusty-64:~$ uptime
18:17:22 up 1 day, 25 min, 1 user, load average: 0.00, 0.01, 0.05
vagrant@vagrant-ubuntu-trusty-64:~$ host udacity.com
udacity.com has address 50.116.54.191
udacity.com mail is handled by 30 ASPMX2.GOOGLEMAIL.com.
udacity.com mail is handled by 30 ASPMX3.GOOGLEMAIL.com.
udacity.com mail is handled by 10 ASPMX.L.GOOGLE.com.
udacity.com mail is handled by 20 ALT1.ASPMX.L.GOOGLE.com.
udacity.com mail is handled by 20 ALT2.ASPMX.L.GOOGLE.com.
vagrant@vagrant-ubuntu-trusty-64:~$ host lwn.net
lwn.net has address 72.51.34.34
lwn.net mail is handled by 10 tex.lwn.net.
vagrant@vagrant-ubuntu-trusty-64:~$ ls
stuff.zip
vagrant@vagrant-ubuntu-trusty-64:~$ █
```

It can be disconcerting to run a command  
and get no output or very little output,



1:23 / 1:35



YouTube





How many items are in the things\_with\_shells directory?

```
Philip@hypercube things_with_shells $ls  
Royal Dutch Shell      egg.md  
clam.rst                pistachio.doc
```

number of items:

4

# Reading Shell Transcripts

input output

input	output
user@host:~\$	cd /tmp/testing/
user@host:testing\$	ls
bears	fish moose squirrels
user@host:testing\$	echo Good St. Moose's Day!
>	oh right, quotes
>	'
	Good St. Mooses Day!
	oh right, quotes
user@host:testing\$	

# Reading Shell Transcripts

input output



```
user@host:~$ cd /tmp/testing/  
user@host:testing$ ls  
bears  fish  moose  squirrels  
user@host:testing$ echo Good St. Moose's Day!  
> oh right, quotes  
> '  
Good St. Mooses Day!  
oh right, quotes
```

The way you do that is to put  
a backslash before the apostrophe.

Mark each line of the transcript as either  
containing user input or command output.



0:51 / 0:59



YouTube

