

# VirtualBox Setup

## CSCE 410 Spring 2021

You need to set up your *VirtualBox* environment for the 410 projects:

<https://www.virtualbox.org/>

If you prefer to use another environment that works for you, it is ok as long as you can run the Bochs hardware emulator. Bochs requires display manipulation, so adopting environments like Vagrant may be tricky. If you succeed with a convenient environment, share with us by posting about it on Piazza.

You can download *VirtualBox* at <https://www.virtualbox.org/wiki/Downloads>



The instructor update their platform to version 6.1.18 in her, but if you have an older version of VirtualBox in your machine, it should work (for example, last Spring the instructor used 6.1.2.)

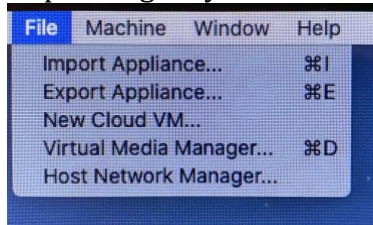
Next, you do the setup, which is quite intuitive. If you have specific questions for your environment, make sure to ask at our [Piazza](#) forum.

We have an Ubuntu image ready for you, available at the course Shared Drive: linux.ova

Now run VirtualBox and import linux.ova as follows:

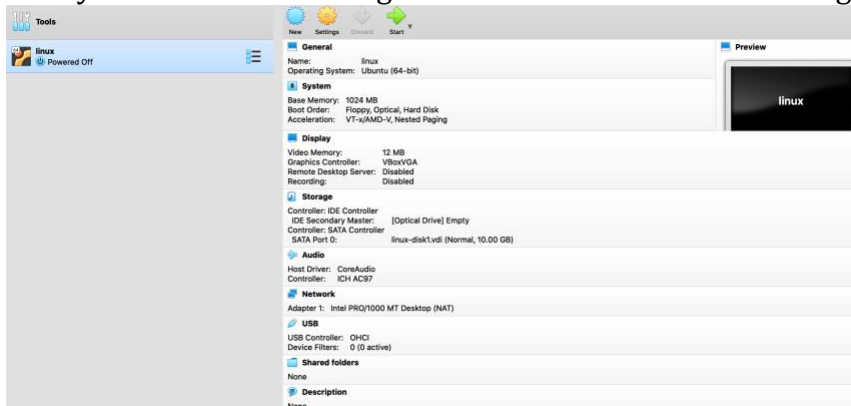
- Download the linux.ova file;

- Go to File -> Import Appliance  
Depending on your environment, it may look like

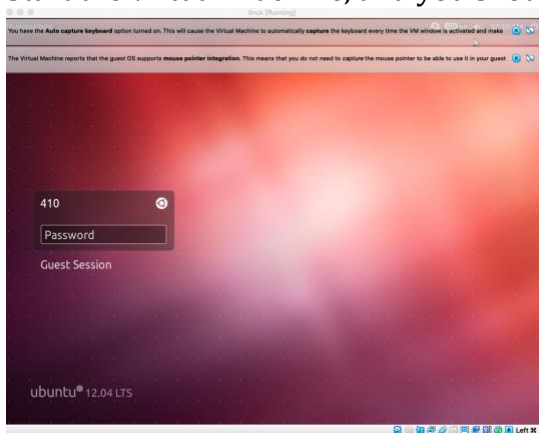


- Provide the location for the linux.ova file you just downloaded;
- Import with default settings.

Now your VirtualBox management screen will look something like the following:



You created a virtual machine (VM) with the Ubuntu Linux environment you need. Start the virtual machine, and you should get something like:



The login password is **guest123**. If you are not able to type in the VM, it is possible that the VirtualBox application is asking for permission from your system to access the mouse or keyboard input. Look for dialog windows asking for permission (it may be hidden under other applications in your desktop), for example it may want to have access to your audio.

Now you have a Linux machine to work where you will be able to modify files, compiler your own new kernel, and run it on an emulator. The next step is to set a shared folder so that you can access, edit, and save files in your machine that are also available within the Linux virtual machine.

## Setting up a Shared Directory

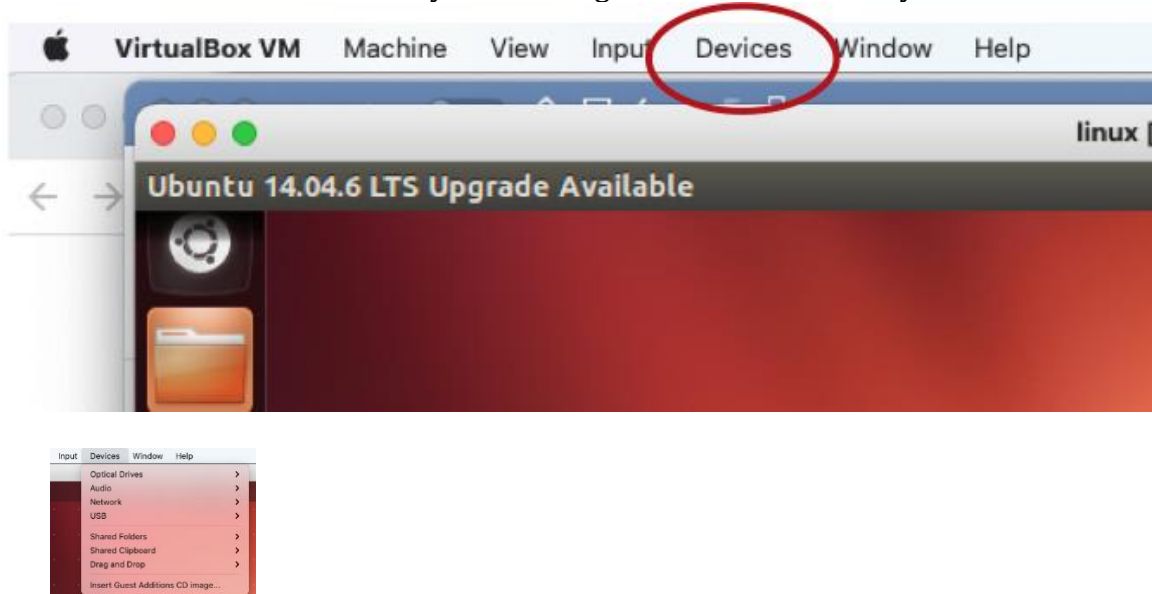
You want to have a folder that you have available from your Linux virtual machine, but that it is actually located elsewhere, so it does not disappear when you kill your virtual machine.

The complete documentation for setting up a folder in *VirtualBox* that is shared with your host operating system (the one actually managing your development hardware) is available at Chapter 4.3 of the [Virtual Box User Documentation](http://download.virtualbox.org/virtualbox/UserManual.pdf) <http://download.virtualbox.org/virtualbox/UserManual.pdf>

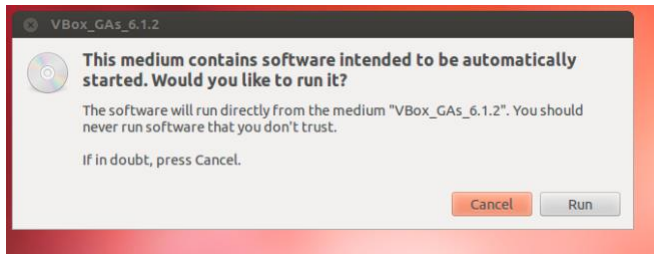
The steps below may be sufficient for you to get your setup ready. You can also find additional guides on the web by searching for “virtual box setup shared folder windows 10” (or whatever environment you have.) Do not hesitate to ask for help through the Piazza forum to learn how other people may have solved the problem you are having.

1. First you need to add the support for shared folders that comes as a Guest Addition to your virtual machine.

Look for the “Devices” menu of your running VirtualBox VM. It may look like:



2. Select “Insert Guest Addition CD image ...” (or something similar, depending on your version), and proceed with the installation. You may get the prompt below; if so, click run and proceed to Step 3.



If you do not receive the automatic prompt above, you may be required to run the Linux additions script manually. The instructor installed VirtualBox in three environments, all MAC OS but different versions, and did not experience problems. Given that VirtualBox support for Windows is more mature, hopefully this part of the setup will work without difficulties.

About errors:

- One error the instructor saw was a window with the error message starting with 'Unable to insert the virtual optical disk', with some text complaining about permissions. This happened in situations where, after importing the appliance, the developer needed to stop working and therefore stopped the machine and restarted it again.  
A path that worked was restarting from scratch: shutdown the virtual machine and ask VirtualBox to remove the virtual machine, deleting all the its files. Then import the machine again from the pristine linux.ova file that is available in the google drive.
- If starting from scratch did not help, try running the scripts manually. The recommendation from previous semesters was the following:

Look for the scripts in /media:

```
guest@TA-virtualbox: ~  
guest@TA-virtualbox:~$ ls /media/VBox_GAs_6.1.18/  
AUTORUN.INF  runasroot.sh          VBoxSolarisAdditions.pkg  
autorun.sh   TRANS.TBL             VBoxWindowsAdditions-amd64.exe  
cert         VBoxDarwinAdditions.pkg  VBoxWindowsAdditions.exe  
NT3x         VBoxDarwinAdditionsUninstall.tool  VBoxWindowsAdditions-x86.exe  
OS2          VBoxLinuxAdditions.run  
guest@TA-virtualbox:~$
```

If you encounter any permissions errors, you may need to set your permissions using "sudo usermod -aG vboxsf guest". For any prompts for a password, enter "guest123".

The instructor believes this is a deprecated recommendation, because it seems group vboxsf does not exist.

Again, do not hesitate in asking for help in Piazza and requesting the instructor to help you.

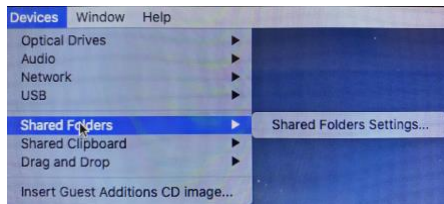
3. Now that you have sharedfolder support in your VM, you need to create the shared folder that you want. There are two possible paths. At any point that things do not work and you think you may have made a mistake, you can simply stop your VM, remove it (and its files), and start again by importing the

appliance linux.ova

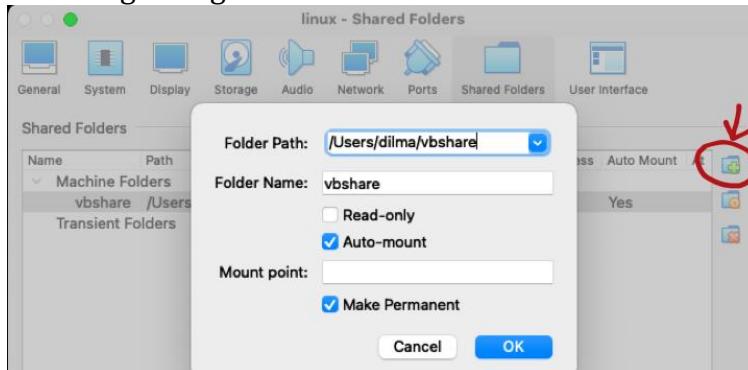
Both paths start with creating a mount point in your Linux VM, i.e., the directory in your file system where you will find the shared folder. You can use any name you like. The example below has “vbshare” as the name of the directory you are creating to mount the shared folder:

```
guest@TA-virtualbox:~$ mkdir vbshare
```

### Path 1: using the support on the VM desktop



Click on the “+” icon on the right and select a path on your host machine with the following configuration:



For mount point, enter the directory you created in your host machine.

Select auto-mount and make permanent. About the name:

- If your host machine is a Mac or a linux, avoid using paths that include symbolic links<sup>1</sup>. The instructor had problems with remote directories (in past semesters trying providers Dropbox.com, drive.google.com, and drive.google.tamu.edu.), but some students stated that it worked well for them. If you, like the instructor, uses a local directory, make sure that you have a process to back up your code regularly (you should be committing your code to GitHub regularly; if you are not backing up constantly, push your changes to the GitHub server very often.)

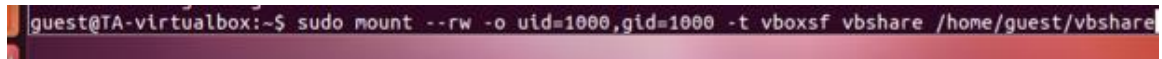
---

<sup>1</sup> The instructor had problems with this and found plenty of reports that the support for symbolic links does not work well. It is not surprising, as we will see when we discuss file systems later in the semester.

- For “Folder name”, do not include a name with spaces. The instructor recalls having problem with symbols (like a dash in vb-share) in the past. For mount point, enter the directory you created above. Select auto-mount and make permanent.

You should now be able to restart your linux VM and, once it comes back, it should be automatically mounted. One way to restart the machine is to kill the window with the guest; it will ask if you want to save the state and then you can go to the VirtualBox console and restart.

It may be that, at restart, you do not have the filesystem for the shared folder mounted in your VM; the mountpoint (/home/guest/vbshare) will be an empty directory, instead of reflecting what you have in the host machine. You need to mount the filesystem manually:



```
guest@TA-virtualbox:~$ sudo mount -o uid=1000,gid=1000 -t vboxsf vbshare /home/guest/vbshare
```

This invocation explicitly mounts as read/write and with user id and user group “guest”.

You are done. Now try it out: create a file in your ~/vbshare and a file in the directory in your machine, and see them appearing on both places. You can add the mount command to your /etc/rc.local in order to have it mount automatically. If you end up having to run it manually, recall that you can use the up/down arrows to cycle through the last commands you run.

## Path 2: using VBoxManage on your host system

If Path 1 did not work, you can try the following:

From the command line, you can create shared folders using VBoxManage, as follows:

```
VBoxManage sharedfolder add "VM name" --name  
"sharename" --hostpath "C:\test"
```

The documentation indicates that there is an option for automount, but there are reports it does not work. You can look at the VirtualBox manual and try.

You now need to mount the shared folder. Automatic mounting (done after boot automatically) requires a particular group to work. For manually mounting, see the picture with the mount invocation above.

- Check Auto-mount and Make Permanent and click the down arrow to configure the path to the folder. If target folder is H:/ Drive the Host machine must be connected to the TAMU network or VPN

## Note:

The projects in this course will all be run through your virtual machine, which should have all the necessary software already installed for you. When you are done with a session, simply close the virtual machine window, and it will prompt you to save its state. When you start again, it will allow you to resume directly where you left off. This will save time navigating directories and you can simply use the linux terminal's history to expedite your work.

Also, depending on your host machine, you may find it beneficial to edit your code in your host machine and use the virtual machine to build and run the software, as the virtual machine may be significantly slower than your host machine.