Q3 FY 2017 Federal SE KSO

DevNet Express Module 0 Walk-Through

Setting up your Dev Environment

By: Mark Nguyen

Note: This document goes through only one method for setting up your Dev environment: using Linux running inside VirtualBox.

February 27, 2017

Table of Contents

Change Log

First draft (February 24, 2017)

Introduction

This document steps through the process for setting up a linux development environment on your laptop. This method was chosen because it is easy to distribute and does not pollute your everyday work laptop. The screen shots in this document were capture on a Macbook. If you are using Windows or Linux, they may look slightly different.

Setup your Hypervisor and OS

Download and install VirtualBox

Download and install VirtualBox for your operating system.

https://www.virtualbox.org/wiki/Downloads

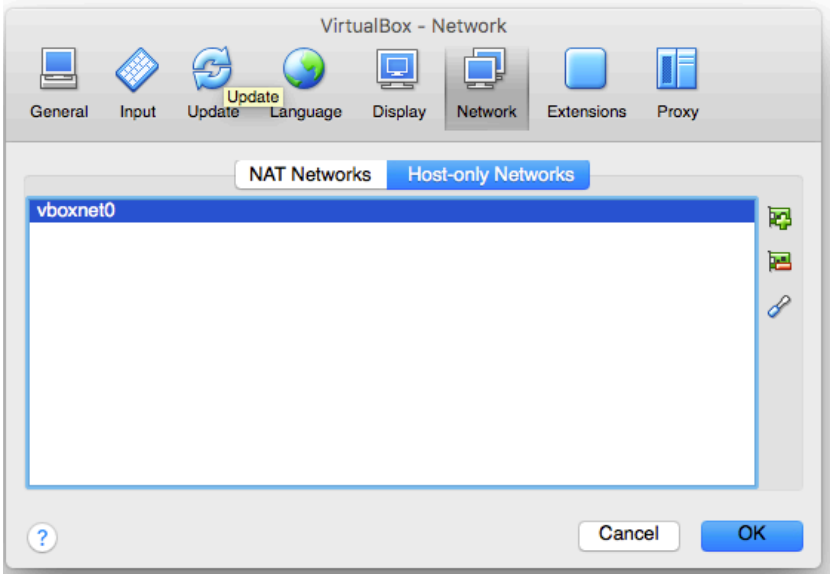
Note: The VirtualBox Extension Pack is not required.

If you already have version 5.0 running, I would upgrade it to the latest 5.0.x version (to avoid potential issues migrating your existing VMs to 5.1.x). For those who don't already have VirtualBox installed, download version 5.1.x.

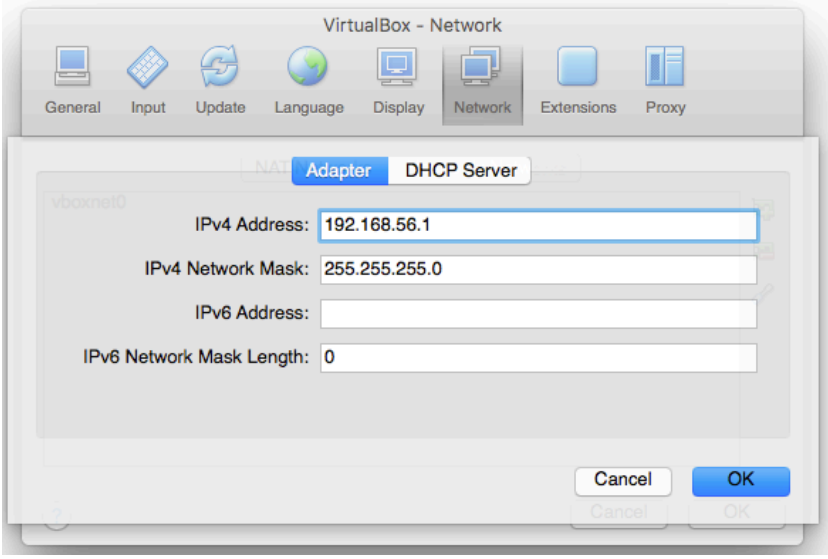
Next, we want to setup a "Host-only" network. This will be used to connect from your host OS to your guest OS.

In VirtualBox, select VirtualBox -> Preferences.

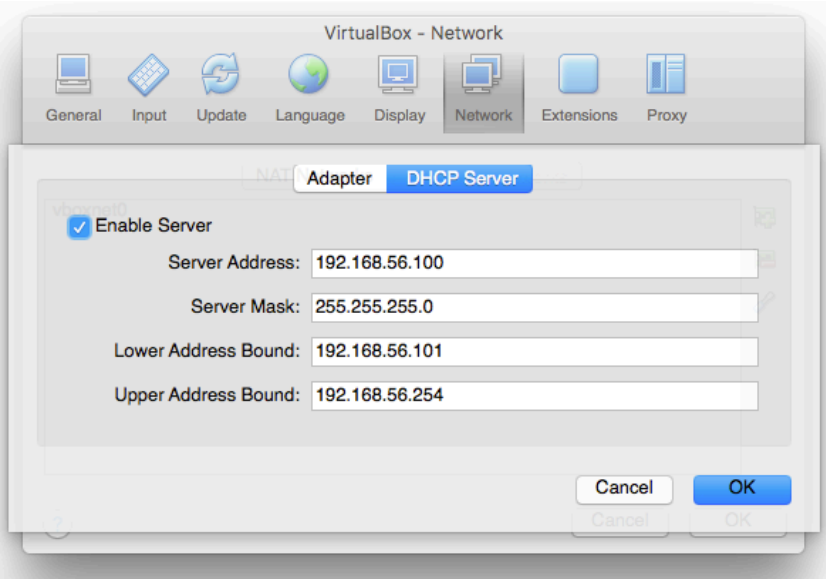
Go to the Network tab and select **Host-only Networks.**



Click on the screwdriver to edit **vboxnet0**. Fill in an IP address and subnet. I used 192.168.56.1 but you can use any private subnet that you're not using.



Now select the DHCP Server tab and ensure the DHCP server is enabled and is populated with a range.



Click OK to accept changes and OK again to save and exit Preferences.

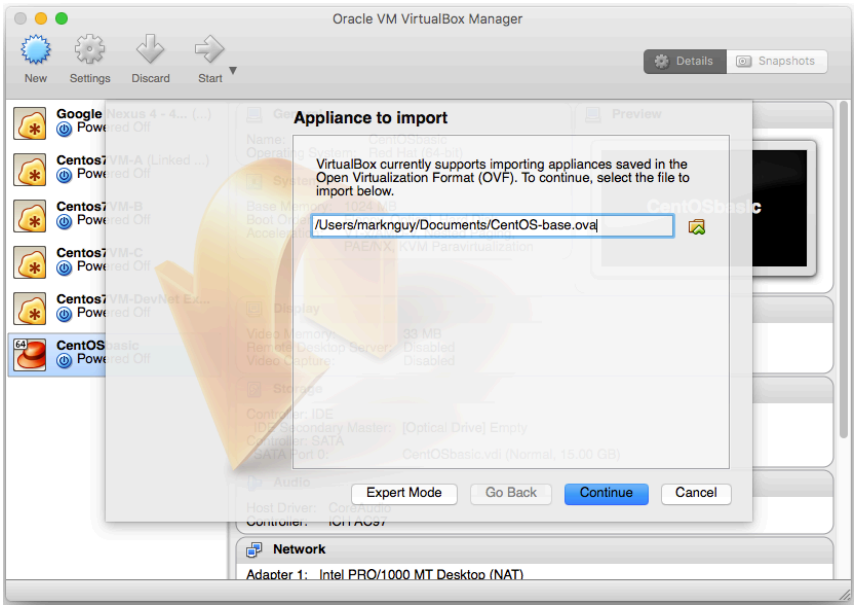
Download and deploy CentOS

I've built a basic CentOS ova for you to download and install. For those who are more adventurous and would like to install from ISO, see Appendix A.

Download pre-built ova from box:

https://cisco.box.com/v/centOS7-basic-marknguy

Once downloaded, in VirtualBox, select File -> Import Appliance. Browse and select the ova you just downloaded:

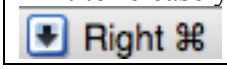


Select Continue and then Import.

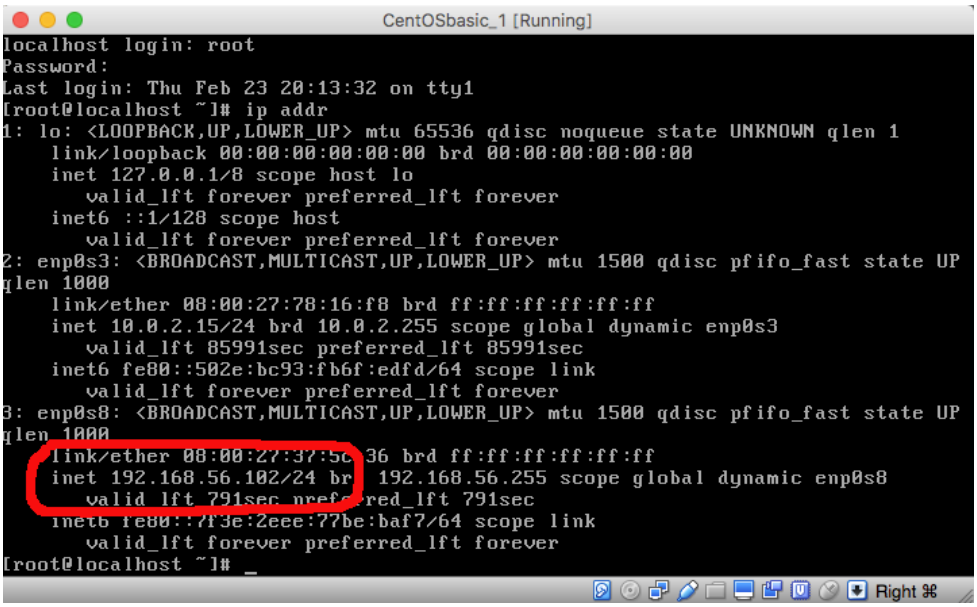
First time Power On and Access

Power on your new VM by highlighting it and selecting the Start 

button. It takes only a few seconds to boot. Login with the username "root" and password "cisco123".

TIP: to release your mouse from the VM, use the "Right-Command" button on a Mac.


Issue the command "ip addr" and note the IP address within the subnet you assigned to the VirtualBox host-only network. In this case, the address is 192.168.56.102.



Alternately, if you are on a Mac, you can issue this command:

```
MARKNGUY-M-908A:~ markngguy$ netstat -rn|grep 192.168.56
192.168.56         link#17            UC              4        0 vboxnet
192.168.56.102     link#17            UHLWIi          1        2 vboxnet
```

This is the address of your guest VM that you can access from your host. Using your favorite terminal client, SSH to it and login:

```
MARKNGUY-M-908A:~ markngguy$ ssh root@192.168.56.102
The authenticity of host '192.168.56.102 (192.168.56.102)' can't be established.
RSA key fingerprint is SHA256:rx3dhjZ3Ad3fIQaqhQHr2Ps89Do86iNQ1m22GOiPUdk.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.56.102' (RSA) to the list of known hosts.
root@192.168.56.102's password:
Last login: Thu Feb 23 20:14:05 2017
[root@localhost ~]#
```

It is much easier to access your new CentOS command line from your terminal client than your VirtualBox window. Things like cut-and-paste work better and resizing your terminal window work better.

Issue the command "`yum –y update`" to upgrade your CentOS system software.

Issue the command "`yum –y install yum-utils`" to enhance yum's capabilities.

Create a New Sudo User

Generally, you don't want to be logged in as root all the time. Let's create your user account with sudo privileges.

1. Use the `adduser` command to add a new user to your system.

   Be sure to replace username with the user that you want to create.

   - `adduser username`
   - Use the `passwd` command to update the new user's password.
     - `passwd username`
   - Set and confirm the new user's password at the prompt.

     ```
     Set password prompts:
     Changing password for user username.
     New password:
     Retype new password:
     passwd: all authentication tokens updated successfully.
     ```

2. Use the `usermod` command to add the user to the `wheel` group.
   - `usermod -aG wheel username`

   By default, on CentOS, members of the `wheel` group have sudo privileges. Logout and login as your new user.

```
[root@localhost ~]# adduser markнguy
[root@localhost ~]# passwd markнguy
Changing password for user markнguy.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
[root@localhost ~]# usermod -aG wheel markнguy
[root@localhost ~]# logout
Connection to 192.168.56.102 closed.
MARKNGUY-M-908A:~ markнguy$ ssh markнguy@192.168.56.102
markнguy@192.168.56.102's password:
[markнguy@localhost ~]$
```

Install Operational Tools on your Host

(from original Module 0). This only lists the required tools for your main host.

Cisco Spark

(Required) Download and install the Cisco Spark client for your platform. Native clients are available for Windows and Mac (and Android and iOS). If on Linux (or if you don't want to install the native client) you can use the Web client.

Postman

(Required) Postman is a tool for testing and exploring REST APIs. Either install a stand-alone version or install it as an extension into Chrome.

Text Editor

(Required) You will need a text editor to edit source code files. This varies greatly across platforms and also personal preferences. These are some recommendations. Just make sure that your text editor of choice saves text files and not some binary format.

- **Windows:** Atom, Sublime Text or Notepad++ are some choices to consider. NotePad++ is a lightweight and free option to consider if you don't have any preference at all.
- **Mac OS X:** Again, Atom and Sublime Text are powerful choices. Another option is TextWrangler.
- **Linux** Again, Atom and Sublime Text and plenty of other options like gedit or Kate.

SSH / SCP

(Required) If you don't have it already (most Linux and Mac OS X users should be good to go) then we also recommend to install a SSH suite. Putty is a popular choice on Windows.

Another powerful Windows tool is WinSCP which integrates nicely into Windows and allows for easy and secure file transfers.

Install Development Tools

Install CentOS Development Tools

This will allow us to compile source code:

```
$ sudo yum -y groupinstall development
```

This will allow us to compile the cryptography package later

```
$ sudo yum -y install openssl-devel
```

Install Python 3

CentOS 7 already comes with python 2.7 installed, but it's been around since 2010. Therefore we are going with Python 3.5.2 (the latest version is 3.6.0, but I like to avoid the first release of anything).

This was the best tutorial I found for installing python 3 (https://www.digitalocean.com/community/tutorials/how-to-install-python-3-and-set-up-a-local-programming-environment-on-centos-7) so I'm copying some of it here for your reference.

We would like to install the most current upstream stable release of Python 3, so we will need to install **IUS**, which stands for Inline with Upstream Stable. A community project, IUS provides Red Hat Package Manager (RPM) packages for some newer versions of select software.

To install IUS, let's install it through `yum`:

- `sudo yum -y install https://centos7.iuscommunity.org/ius-release.rpm`

Once IUS is finished installing, we can install Python 3.5.3:

- `sudo yum -y install python35u-3.5.3`

When the installation process of Python is complete, we can check to make sure that the installation was successful by checking for its version number with the `python3.5` command:

- `python3.5 -V`

With Python 3.5.3 successfully installed, we will receive the following output:

```
Output
Python 3.5.3
```

Install PIP

We will next install **pip**, which will manage software packages for Python:

- `sudo yum -y install python35u-pip`

The `pip` tool is the Python package manager that interfaces with the Python Package index PyPI. We recommend that you use it to manage the installation of Python packages. A Python package is a collection of Python modules. Importing modules can be useful because they provide additional functionality to the Python environment. We will use **pip** to install and manage programming packages we may want to use in our development projects. You can install Python packages by typing:

- `pip3.5 install package_name`

Here, `package_name` can refer to any Python package or library, such as Django for web development or NumPy for scientific computing. So if you would like to install NumPy, you can do so with the command `pip3.5 install numpy`.

Install Python Libraries

Next, we will need to install the IUS package **python35u-devel**, which provides us with libraries and header files we will need for Python 3 development:

- `sudo yum -y install python35u-devel`

Among the libraries in the python35u-devel package is the **pyvenv** package that we will use to set up a virtual environment for our development projects in the next step.

Setting Up a Virtual Environment

Now that we have Python installed and our system set up, we can go on to create our programming environment with pyvenv. Pyvenv comes with Python 3.5.3. In the original Module0, we used virtualenv. Since pyvenv is native to Python 3.5, it makes sense to use the tool that was created by the Python developers.

Virtual environments enable you to have an isolated space on your computer for Python projects, ensuring that each of your projects can have its own set of dependencies that won't disrupt any of your other projects.

Setting up a programming environment provides us with greater control over our Python projects and over how different versions of packages are handled. This is especially important when working with third-party packages.

You can set up as many Python programming environments as you want. Each environment is basically a directory or folder in your computer that has a few scripts in it to make it act as an environment.

You can create an environment by running the following command:

- `pyvenv-3.5 mycode`

Essentially, this command creates a new directory (in this case called mycode) that contains a few items that we can see with the `ls` command:

```
bin include lib lib64 pyvenv.cfg
```

Together, these files work to make sure that your projects are isolated from the broader context of your local machine, so that system files and project files don't mix. This is good practice for version control and to ensure that each of your projects has access to the particular packages that it needs.

To use this environment, you need to activate it, which you can do by typing the following command that calls the **activate** script in the `bin` directory:

- `source mycode/bin/activate`

Your prompt will now be prefixed with the name of your environment, in this case it is called mycode:

```
[marknguy@localhost ~]$ pyvenv-3.5 mycode
[marknguy@localhost ~]$ source mycode/bin/activate
(mycode) [marknguy@localhost ~]$
```

This prefix lets us know that the environment mycode is currently active, meaning that when we create programs here they will use only this particular environment's settings and packages.

Note: Within the virtual environment, you can use the command `python` instead of `python3.5`, and `pip` instead of `pip3.5` if you would prefer. If you use Python 3 on your machine outside of an environment, you will need to use the `python3.5` and `pip3.5` commands exclusively.

Your virtual environment is ready to use.

Add additional packages to the virtual environment

Now that we have our virtual environment set up, let's install some additional packages for class.

```
(mycode) [marknguy@localhost ~]$ pip install requests netaddr argparse pyang uniq
Collecting requests
  Downloading requests-2.13.0-py2.py3-none-any.whl (584kB)
    100% |████████████████████████████████| 593kB 1.4MB/s
Collecting netaddr
  Downloading netaddr-0.7.19-py2.py3-none-any.whl (1.6MB)
    100% |████████████████████████████████| 1.6MB 781kB/s
Collecting argparse
  Downloading argparse-1.4.0-py2.py3-none-any.whl
Collecting pyang
  Downloading pyang-1.7.1-py2.py3-none-any.whl (300kB)
    100% |████████████████████████████████| 307kB 2.3MB/s
Collecting uniq
  Downloading uniq-1.3.2.34-py3-none-any.whl (601kB)
    100% |████████████████████████████████| 604kB 1.6MB/s
Installing collected packages: requests, netaddr, argparse, pyang, uniq
Successfully installed argparse-1.4.0 netaddr-0.7.19 pyang-1.7.1 requests-2.13.0 uniq-
1.3.2.34


(mycode) [marknguy@localhost ~]$ pip install ncclient
Collecting ncclient
  Downloading ncclient-0.5.3.tar.gz (63kB)
    100% |████████████████████████████████| 71kB 1.1MB/s
Requirement already satisfied: setuptools>0.6 in ./mycode/lib/python3.5/site-packages (from
ncclient)
Collecting paramiko>=1.15.0 (from ncclient)
  Downloading paramiko-2.1.2-py2.py3-none-any.whl (172kB)
```

```
    100% |████████████████████████████████| 174kB 1.8MB/s
Collecting lxml>=3.3.0 (from ncclient)
  Downloading lxml-3.7.3-cp35-cp35m-manylinux1_x86_64.whl (7.1MB)
    100% |████████████████████████████████| 7.1MB 198kB/s
Collecting six (from ncclient)
  Downloading six-1.10.0-py2.py3-none-any.whl
Collecting pyasn1>=0.1.7 (from paramiko>=1.15.0->ncclient)
  Downloading pyasn1-0.2.2-py2.py3-none-any.whl (51kB)
    100% |████████████████████████████████| 61kB 4.7MB/s
Collecting cryptography>=1.1 (from paramiko>=1.15.0->ncclient)
  Downloading cryptography-1.7.2.tar.gz (420kB)
    100% |████████████████████████████████| 430kB 1.8MB/s
Collecting idna>=2.0 (from cryptography>=1.1->paramiko>=1.15.0->ncclient)
  Downloading idna-2.2-py2.py3-none-any.whl (55kB)
    100% |████████████████████████████████| 61kB 4.8MB/s
Collecting cffi>=1.4.1 (from cryptography>=1.1->paramiko>=1.15.0->ncclient)
  Downloading cffi-1.9.1-cp35-cp35m-manylinux1_x86_64.whl (398kB)
    100% |████████████████████████████████| 399kB 1.7MB/s
Collecting pycparser (from cffi>=1.4.1->cryptography>=1.1->paramiko>=1.15.0->ncclient)
  Downloading pycparser-2.17.tar.gz (231kB)
    100% |████████████████████████████████| 235kB 2.4MB/s
...
Installing collected packages: cryptography, paramiko, lxml, ncclient
  Running setup.py install for cryptography ... done
  Running setup.py install for ncclient ... done
Successfully installed cryptography-1.7.2 lxml-3.7.3 ncclient-0.5.3 paramiko-2.1.2


(mycode) [marknguy@localhost ~]$ pip list
DEPRECATION: The default format will switch to columns in the future. You can use --
format=(legacy|columns) (or define a format=(legacy|columns) in your pip.conf under the
[list] section) to disable this warning.
cffi (1.9.1)
cryptography (1.7.2)
idna (2.2)
ipaddress (1.0.18)
lxml (3.7.3)
ncclient (0.5.3)
netaddr (0.7.19)
paramiko (2.1.2)
pip (9.0.1)
pyang (1.7.1)
pyasn1 (0.2.2)
pycparser (2.17)
requests (2.13.0)
setuptools (28.8.0)
six (1.10.0)
uniq (1.3.2.34)
```

Final Steps

Install Git

(Required) Git v1.8.x is already installed as part of the CentOS 7.

Get a Spark Account

(Required) Throughout the learning track you will use Cisco's instant messaging tool called Cisco Spark. To use Cisco Spark you need a Spark account. Register with Cisco Spark if you do not already have a Spark account. It's a free service. You only need an email address to register.

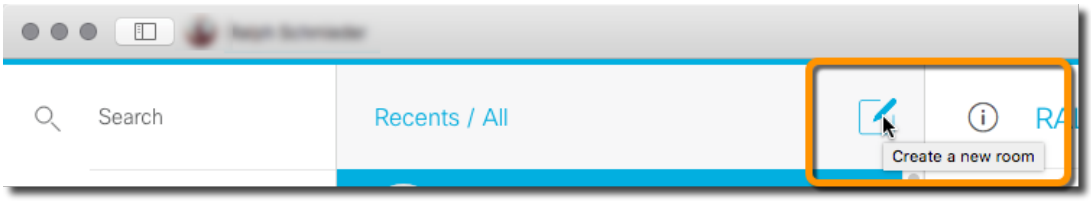
1. Go to https://www.ciscospark.com
2. Fill out the registration info

Create a Spark Room

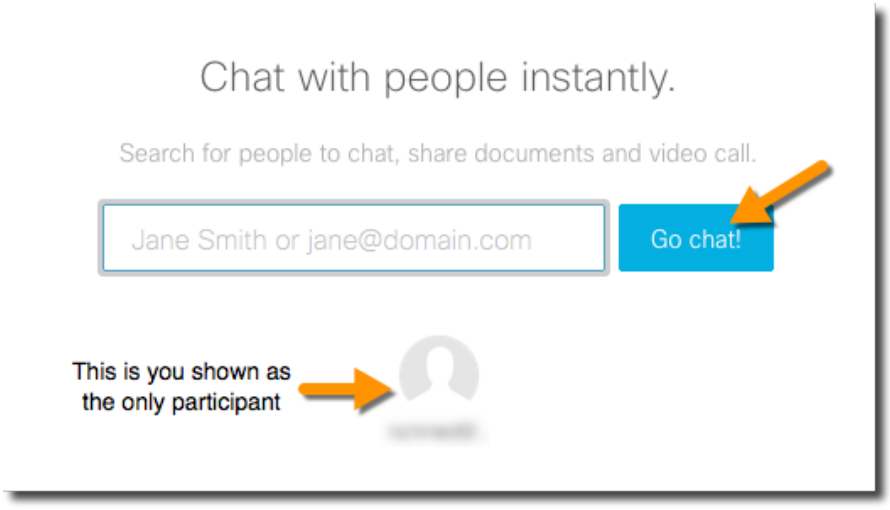
Create a test room in Spark where you can send yourself messages (unless told otherwise, in a hands-on event you will be provided with a specific event room name where everybody posts messages).

To create a Spark room:

1. Open your Spark client or the Spark Web client
2. Create a new room



3. Add your email address, followed by 'Return'



4. Click 'Go chat!' to create the room
5. The room will only be started when you send a message to it.
6. Type a message. The room shows as 'Untitled'

7. Click the title and change the room name to something memorable such as 'LetMySPARKRoomFly'

Congratulations, you have a Spark room! You can use this room later when you perform tests against the Spark API.

Get a GitHub Account

(Optional) While this is not strictly required, we recommend that you join GitHub.

1. Go to https://github.com/join
2. Fill out registration info

You are not required to have a Github account. The repository that we use in this track is a public repository, which does not require that you have a Github account.

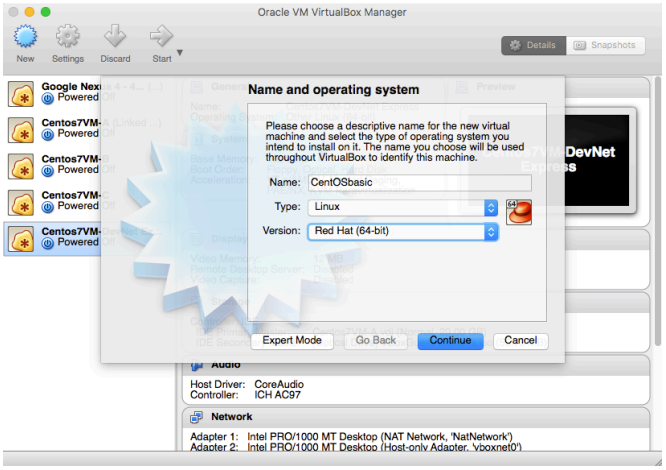
However, a Github account can be useful to you because you will be able to create your own code repositories and save your personal code there. For example, you'll be able to store the source code that you create during the labs to your GitHub account for your use later.

Appendix A: Install CentOS from ISO

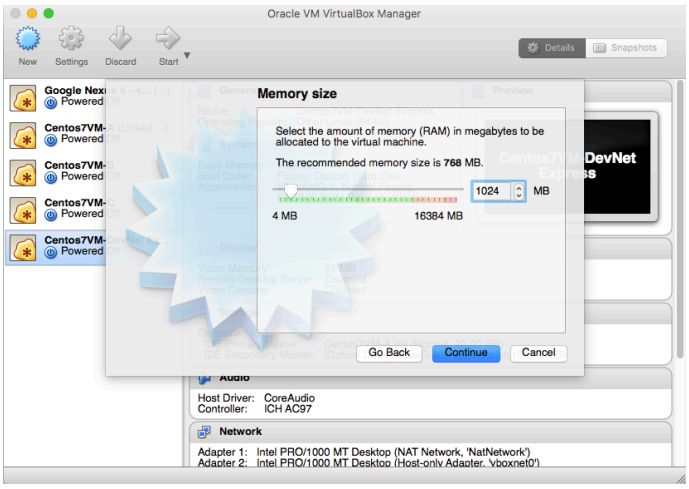
Download the minimal CentOS 7 image from one of these repositories:

http://isoredirect.centos.org/centos/7/isos/x86_64/CentOS-7-x86_64-Minimal-1611.iso

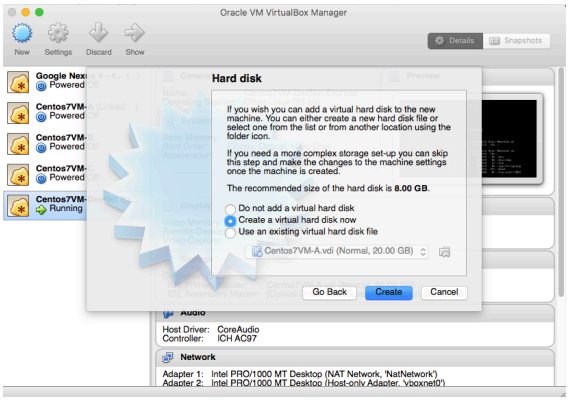
In VirtualBox, select **New** from the top left

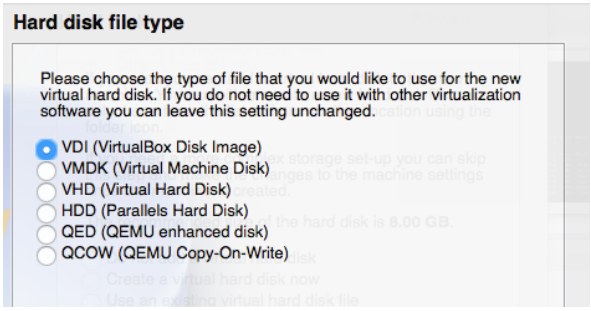


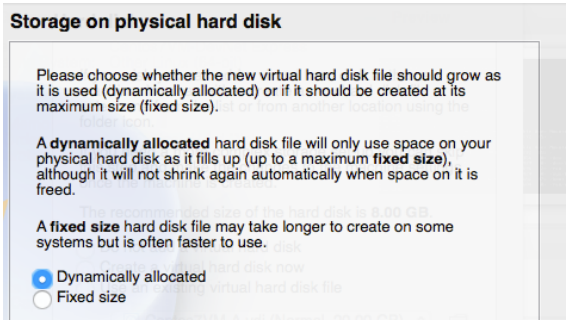
Follow the following screen shots to deploy your VM (please note that this was for VirtualBox version 5.0.32. Your windows may be slightly different if you are using a different version).

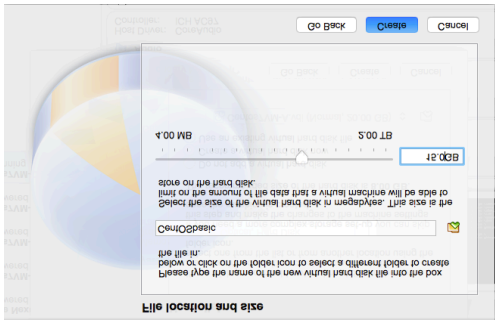


Allocate 15GB of storage:









Pre-built OVA

If you have trouble, you can always get the pre-built OVA for CentOS with Module0 completed and ready to start developing:

https://cisco.box.com/v/centOS7-complete-marknguy