# Steps to complete the Tech Challenge

AUTHOR: Roberto Aarón Herrera Reyna
DATE: 25 / JUL /2018
VERSION: 1.0

REVIEWER : Arturo  Plauchu

DATE:

You can also see the online version of these document in Google Drive:
https://docs.google.com/document/d/1n2DJziuf2PPqxj4YZTYF2YOhK-QtAjWF4kPC7xDYB-k/edit?usp=sharing

You can following my update in the this repository in GitHub:
https://github.com/RobeHerrera/ImageTransferTCP

Here are some of the steps,  that I use to complete the Tech Challenge:

1.- Download a virtual image of Ubuntu. 18.xx
    1.1 A- Make a bootable USB
        B- Make a virtual machine

2.- Install Docker with the following commands
https://docs.docker.com/install/linux/docker-ce/ubuntu/#set-up-the-repository

2.1 - Uninstall

```
sudo apt-get remove docker docker-engine docker.io
```

2.2 - Set up the repository (update apt package index)

```
sudo apt-get update
```

2.3 -Install packages to allow `apt` to use a repository over HTTPS

```
sudo apt-get install \
apt-transport-https \
ca-certificates \
curl \
software-properties-common
```

NOTE: if is necessary first run `sudo apt-get - f install` to install the dependencies

2.4 - Add Docker's official GPG key:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

2.5 - Use the following command to set up the **stable** repository

```
sudo add-apt-repository \
"deb [arch=amd64] https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) \
stable"
```

2.6 - update apt package index

```
sudo apt-get update
```

2.7 - Install Docker CE latest

```
sudo apt-get install docker-ce
```

2.8 - Verify the installation

```
sudo docker run hello-world
```

```
sudo docker version
```

2.9.- Docker added in sudo mode

```
$ sudo groupadd docker
$ sudo usermod -aG docker $USER
$ docker run hello-world (just to verifyx)
```

3.0.- Install git

https://www.howtoforge.com/tutorial/install-git-and-github-on-ubuntu-14.04/

```
sudo apt-get install git
```

3.1.- Configuring GitHub

```
git config --global user.name "user_name"
```

```
git config --global user.email "email_id"
```

3.2.- Creating a local repository

```
git init Mytest
```

And navigate to it `cd Mytest`

3.3.- Create a README file
`gedit README` and write the description ex: `This is git repo`

3.4.- Creating a local repository
`git add README` and `git add sample.c`

3.5.- Commiting changes made to the index

```
git commit -m "some_message"
```

3.6.- Commiting changes made to the index

```
git commit -m "some_message"
```

3.7.- Creating a repository on GitHub

```
git remote add origin https://github.com/user_name/Mytest.git
```

3.8.- Pushing files in local repository

```
git push origin master
```

Learning how to Using Docker

4.0 - Go to https://github.com/docker/labs/tree/master/beginner and follow the instructions in readme.md
Test your installation `docker run hello-world`

4.1 - To get the image of a Linux Alpine -> `docker pull alpine`
4.2 -You can see all the images of your machine using the command -> `docker images`
4.3 -Now we have alpine in our system we can use command like -> `docker run alpine echo "hello from alpine"`
4.4 -Try something else -> `docker run -it alpine /bin/sh`
4.5 -To see the process of docker use -> `docker ps`
4.4 -Write the file and build -> `docker build ~/Documents/flask-app/`
`Files:`

- `app.py`
- `requirements.txt`

- `templates/index.html`
- `Dockerfile`

Using Docker - Machine
https://docs.docker.com/machine/get-started/#create-a-machine
http://www.macadamian.com/2017/01/24/docker-machine-basic-examples/

5.0 - Install Docker- Machine

5.1-Install onLinux- `$`

```
base=https://github.com/docker/machine/releases/download/v0.14.0 &&
  curl -L $base/docker-machine-$(uname -s)-$(uname -m) >/tmp/docker-machine &&
  sudo install /tmp/docker-machine /usr/local/bin/docker-machine
```

5.2 - Verify the installation:

```
$ docker-machine version
```

5.3 - List all the machines in Docker:

```
$ docker-machine ls
```

5.3 - Create Machine

```
$ docker-machine create --driver virtualbox default
```

5.4- If Virtual Box was not installed so installed with ubuntu software center before run the above command. You could installed Ubuntu Software.

```
Go to ubuntu Software and search for virtualbox
```

5.5- Run again the create machine command, wait for a while and next to get the environment commands use the following command:

```
docker-machine env default
```

5.6- Connect your shell to the new machine.

```
$ eval "$(docker-machine env default)"
```

5.7- Use docker run to download and run busybox. And make a simple echo

```
$ docker run busybox echo hello world
```

5.8- Get the host IP address

```
$ docker-machine ip default
```

5.9- Run a Nginx webserver in a container with the following command, it is only and example of how easy we can create a web server:

```
$ docker run -d -p 8000:80 nginx
```

5.10- When we call the ip of the server with the port 8000 we have a welcome page of Nginx, Ex:

```
$ curl $(docker-machine ip default):8000
```

5.11- Start and Stop  Machines, remember that default is the name of the VM

```
$ docker-machine stop default
$ docker-machine start default
```

5.12- Start and Stop  Machines, remember that default is the name of the VM

```
$ docker-machine stop default
```

5.13-  To know all the functions we could use the command:

```
$ docker-machine help
```

5.14- To go to console with ssh protocol, use the command:

```
$ docker-machine ssh default
```

5.15- To copy files from the local to the remote or virtual machine use:

```
$ docker-machine scp
/home/rohe/Documents/ImageTransferTCP/Server/bin/Debug/Server
docker@default:/home/docker
```

Remember to be in you local machine not in the VM.

**NOTE to how to use the SCP command:**

Copy the file "foobar.txt" from a remote host to the local host

```
$ scp your_username@remotehost.edu:foobar.txt /some/local/directory
```

Copy the file "foobar.txt" from the local host to a remote host

```
$ scp foobar.txt your_username@remotehost.edu:/some/remote/directory
```

5.17 - To see where is you virtual machine use the command:

```
$ docker-machine inspect default
In this case is in /home/rohe/.docker/machine
```

IP header with all the fields

https://nmap.org/book/tcpip-ref.html

Simple TCP example

https://docs.docker.com/machine/get-started/#create-a-machine

6.0 - Run the Simple Demo TCP

https://github.com/samehkamaleldin/socket.cpp

Brief explanation of TCP protocol

http://www.bogotobogocom/cplusplus/sockets_server_client.php

https://www.geeksforgeeks.org/socket-programming-cc/

Images through TCP protocol

https://stackoverflow.com/questions/15445207/sending-image-jpeg-through-socket-in-c-linux

https://stackoverflow.com/questions/33783470/sending-picture-via-tcp#

6.1 - Create a environment in Code::Blocks to easy debugging and running the program.

6.2 - Your binaries should be copy to the virtual machine, use the command describe above SCP.

6.3 - Enter to your VM and run the server and VM client, confirm that the machines could communicate with each other.

6.4 - Port and how to kill process:

To list all the process use:

```
$ ps -a
$ top
```

To kill a process use the PID example:

```
$ ps -3612
```

To list the ports in use and the use the kill PID to end the port use, use the command:

```
$ lsoft -i
```

Or install the package `$ sudo apt-get install procinfo` and the use the command `$ sudo socklist`

**How to set the UserComment with exiftool command**

```
exiftool -m -UserComment="Comentario desde Robert" DSC_0001.JPG
        ->1 image files updated
exiftool -UserComment DSC_0001.JPG
        ->User Comment                  : Comentario desde Robert
```

**RESOURCES :**

O edit the user comment in Exif image

http://www.exiv2.org/doc/exifcomment_8cpp-example.html

IPv4 and IPv6

https://www.ibm.com/support/knowledgecenter/ssw_ibm_i_72/rzab6/xacceptboth.htm

**https://www.ibm.com/developerworks/aix/library/au-endianc/index.html**

```
#define LITTLE_ENDIAN 0
#define BIG_ENDIAN     1

int endian() {
   int i = 1;
   char *p = (char *)&i;

   if (p[0] == 1)
      return LITTLE_ENDIAN;
   else
      return BIG_ENDIAN;
}
```

**https://en.wikipedia.org/wiki/Endianness**

```
/* C function to change endianness for byte swap in an unsigned 32-bit integer */

uint32_t ChangeEndianness(uint32_t value)
{
    uint32_t result = 0;
    result |= (value & 0x000000FF) << 24;
    result |= (value & 0x0000FF00) << 8;
    result |= (value & 0x00FF0000) >> 8;
    result |= (value & 0xFF000000) >> 24;
    return result;
}
```

***OBSERVATIONS:***

- IP Address could be change, this address is asing to VirtualBox
- I need to rewrite some part of the code to pass arguments to assign the port and the address.
- Change the compilation to Release instead of Debug.

- No update of the boot2docker.iso, this file was excluded in the repo. Actually maybe even the image of the VM should be not in the repo.
- The server and the client only support one operation at the time, received the text or send the image,  one per run the program, the issue seems to be the connection process to communicate the server and the client.

***ERRORS***

**Warning: GDB: Failed to set controlling terminal: Operation not permitted [NOT SOLVE]**

- In the console I have the error, in code blocks  when I tried to debug
- I tried to, install other gdb, restart the machine and code blocks

**Error in switch case [error] jump to case label [-fpermisive] [SOLVE]**

Buenas, el problema se debe a la declaración de variables dentro de un case. Si quieres declarar variables en un case tienes que usar las llaves {} para asegurar que el alcance (*scope*) de estas variables se limita a ese case.

***NOTES:***

***To manage the network order and  host order***

We use the following command:

htons() -> from host order to network order

ntohs() -> from network order to host order

Example:

PORT: 12002 -> 0x2E E2

htons(PORT) -> 0xE2 2E

Convert host address from number-and-dots notation to binary data in network byte order
inet_addr()

- IP Server: tcp://192.168.99.100:2376
- IP Client:   tcp://192.168.99.101:2376
- The command ->  exiftool 1.jpg
- ExifTool Version Number       : 10.80
- File Name                     : 1.jpg
- Directory                     : .
- File Size                     : 6.2 kB
- File Modification Date/Time   : 2018:07:27 10:27:49-05:00
- File Access Date/Time         : 2018:07:28 23:16:22-05:00
- File Inode Change Date/Time   : 2018:07:28 23:14:55-05:00
- File Permissions              : rw-rw-r--
- File Type                     : JPEG
- File Type Extension           : jpg
- MIME Type                     : image/jpeg
- JFIF Version                  : 1.01
- Resolution Unit               : None
- X Resolution                  : 1
- Y Resolution                  : 1
- Image Width                   : 128
- Image Height                  : 128
- Encoding Process              : Baseline DCT, Huffman coding
- Bits Per Sample               : 8
- Color Components              : 3
- Y Cb Cr Sub Sampling          : YCbCr4:2:0 (2 2)
- Image Size                    : 128x128

- Megapixels : 0.016

-

***<u>DOUBTS</u>*** **:**

-std::istringstream arg_stream(argv[1]); // initialize string stream from argument [**Solved**] Is converted to integer the string gives in the argument.
http://www.cplusplus.com/reference/sstream/istringstream/istringstream/

-How works the print OnMessage Function that prints the message, [**Solved**] pointer to function, and it gives the argument in the function call on the parameter of OnMessage function.