Doogie Software Setup

A repository with software setup instructions for Doogie Mouse robot.

Introduction

This repository will help you on the setup of the Raspberry Pi Zero W v1.1 which is used in the Doogie Mouse robot. Theses instructions assume that you have already the Raspberry peripherals such as mouse, keyboard and monitor connected on it. See the hardware_instructions or oficial Raspberry instructions for more informations.

These instructions package has been made under Ubuntu 16.04 LTS.

How to Read?

The notes are more or less in chronological order, hence start from top the bottom. Commands are prefixed with the system on which the commands needs to be run:

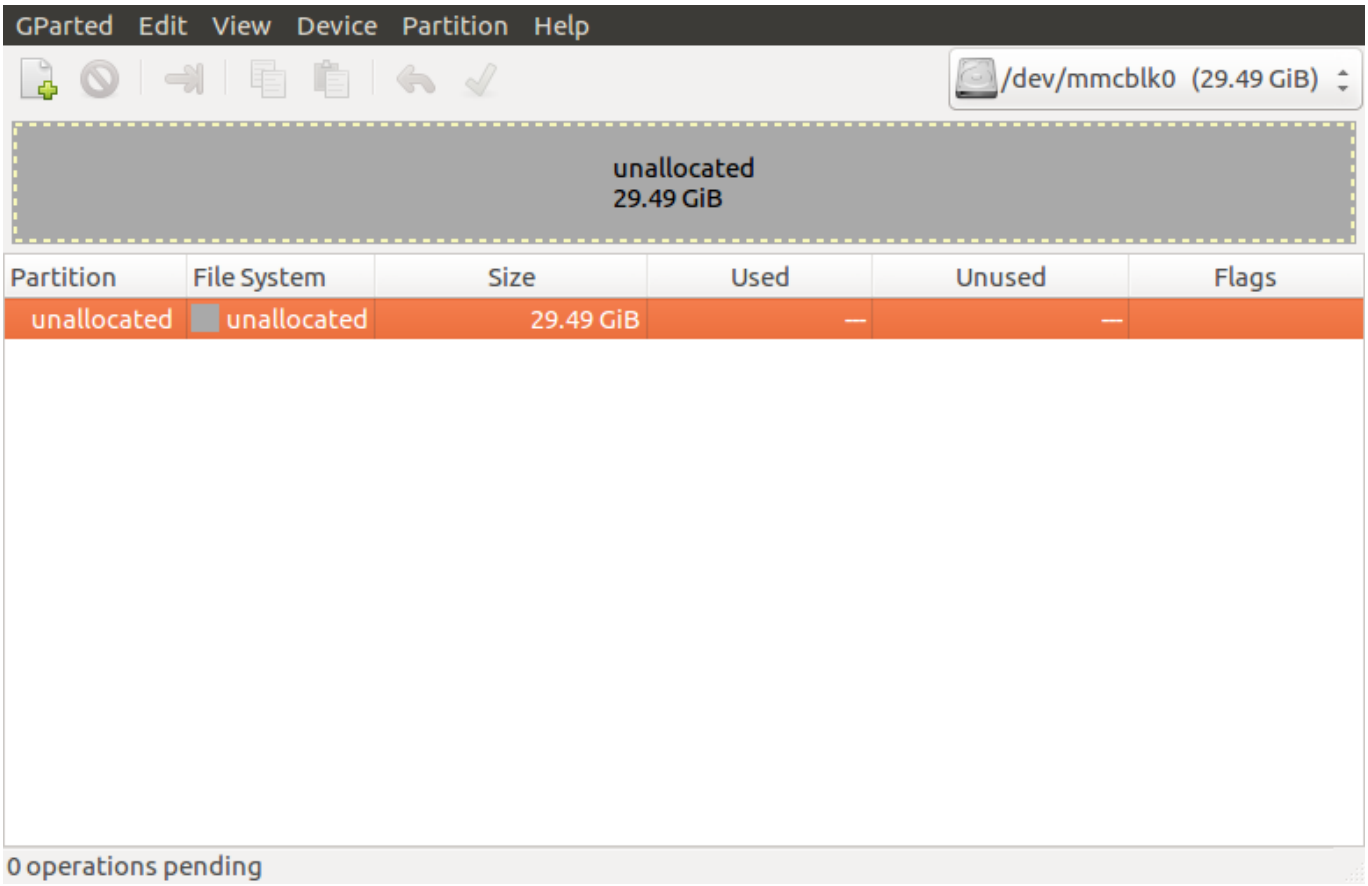
- `HOST:~$` commands executed on your machine
- `RPI:~$` commands executed on the Raspberry Pi

Prepare microSD card

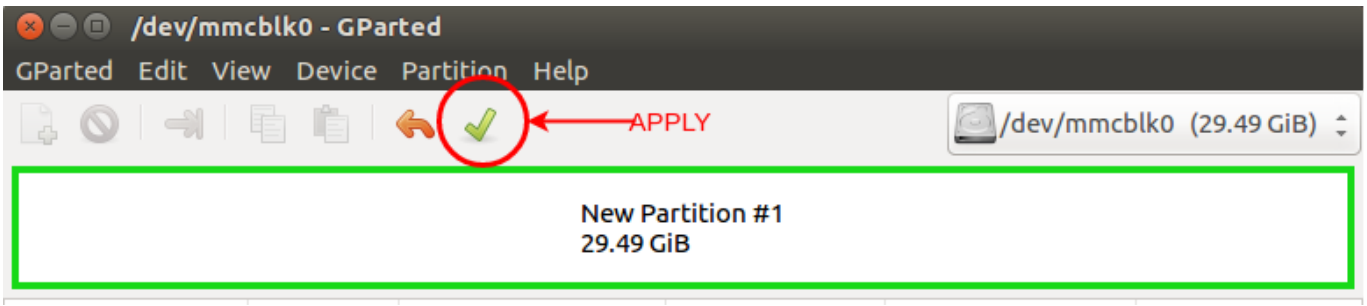
The first step is to format the microSD card. If you are using an old microSD card, remember to do the backup of the media or you will lost all data. To format the card, install and run the GParted (GNOME Partition Editor):

```
HOST:~$ sudo apt-get install gparted
HOST:~$ sudo gparted
```

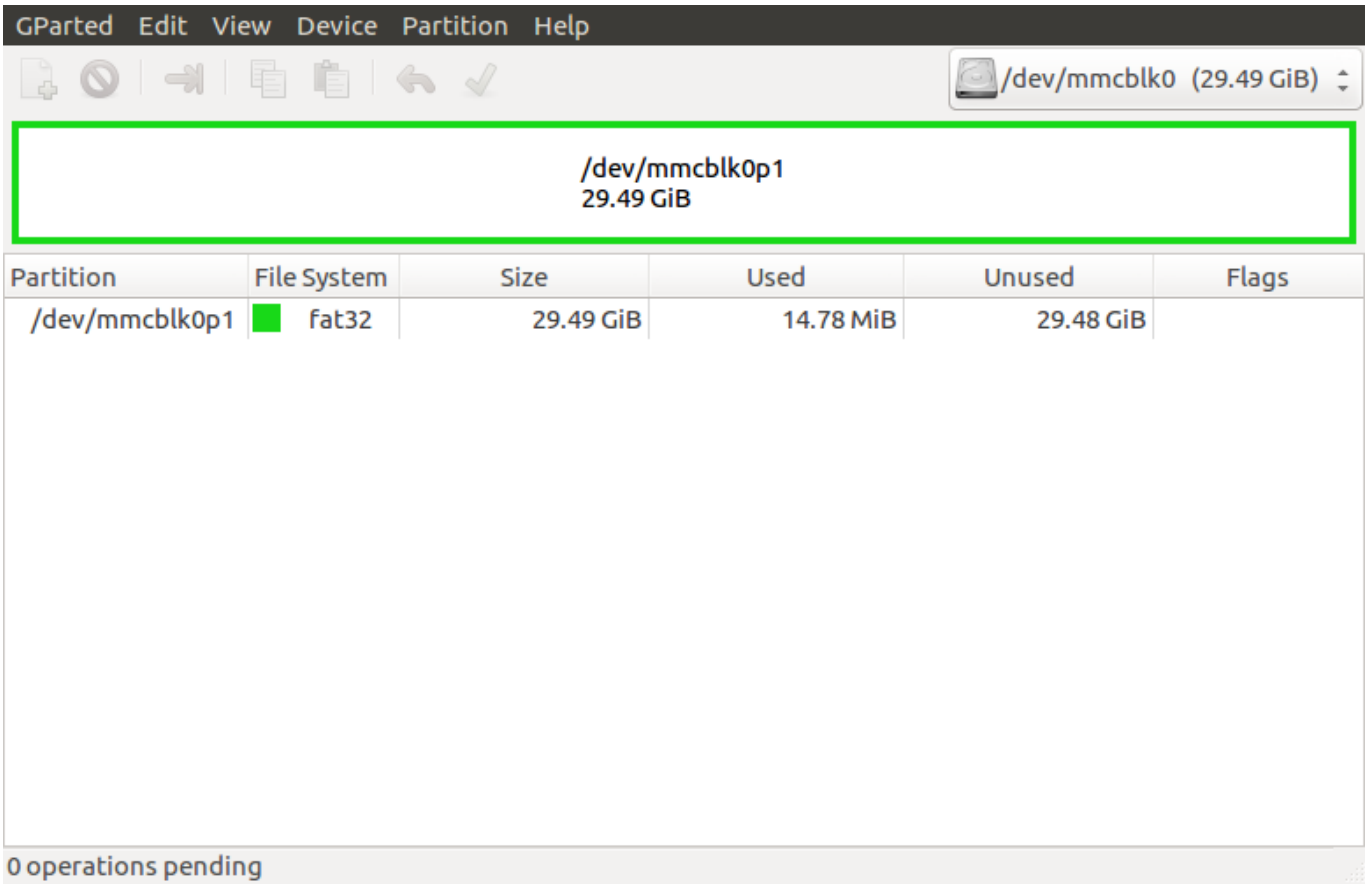
Delete all partitions, if any. You will see the image below.



After that, create a new partition: `right-click->New`. Create the partition with the hole space. In the `File System`, choose `FAT32`, then click in add button and `Apply All Operation`.



Then the gparted will be like this:



See Installing operating system images on Linux for more informations. To finish, unplug and plug the card in your computer.

Raspbian OS Installation

To begin, download the last version of the Raspbian Jessie OS. In our case, download the *Raspbian Jessie Full* version and unzip it.

> If you want download the image using torrent, Ubuntu distribution has a native torrent client called Transmission BitTorrent Client

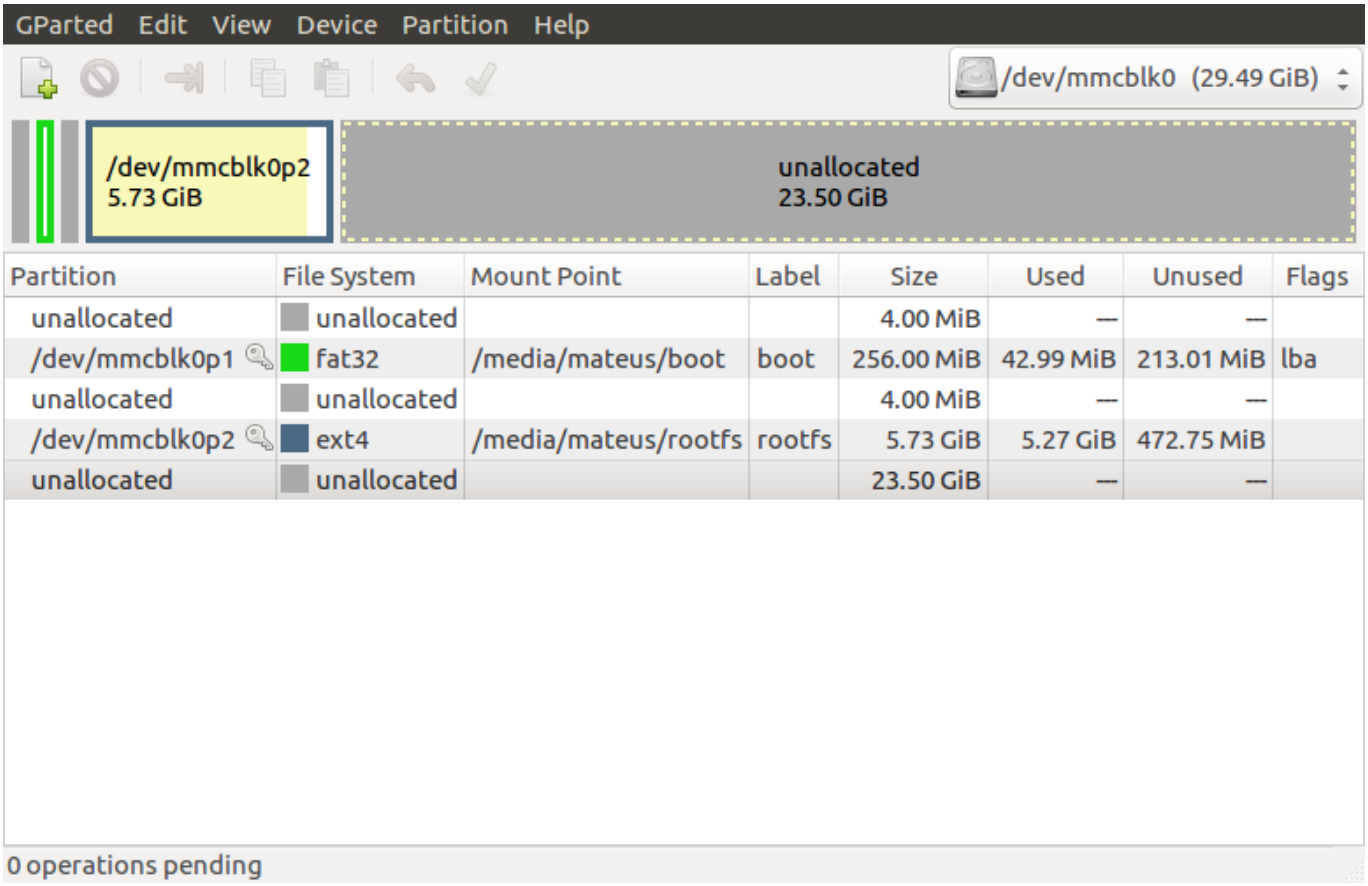
To copy the image to the microSD card

```
HOST:~$ sudo umount /dev/mmcblk0p1
HOST:~$ cd IMG_FOLDER
HOST:~$ sudo dd bs=4M status=progress if=IMG_FILE_NAME.img of=/dev/mmcblk0
```

In the end of the process, you'll see

```
6425673728 bytes (6,4 GB, 6,0 GiB) copied, 253,09 s, 25,4 MB/s
1534+0 records in
1534+0 records out
6434062336 bytes (6,4 GB, 6,0 GiB) copied, 335,209 s, 19,2 MB/s
```

Again, unplug and plug the card in your computer. To check if the process occurred correctly, open the gparted and you'll see the partitions of the card as the image below.



Remove the microSD card and then plug into the Raspberry.

Connect your Raspberry Pi

Make sure at least the keyboard and the monitor are connected to the Raspberry Pi. Now, power on the board using its power pins (5 V and GND) or the USB power port.

Finish the setup

When you start your Raspberry Pi for the first time, some configuration should to be made. Follow the steps below to set up Raspberry:

1. On the display upper left corner, click in the Raspberry ícon
2. Go to `Preferences >> Raspberry PI Configuration`
3. On `System` window, change the fields:
    - `Password` (current is raspberry). Recommended `doogiemouse` for new password
    - `Hostname`. Recommended `doogie-mouse`. **Note:** If multiples Doogie Mouse robots will be used, we recomend use a different second name, e.g., `doogie-brain` for the first robot and for the second robot, `doogie-pinky`. Feel free to choose the best names...
4. On `Interfaces`, enables SSH and I2C
5. On `Localisation`
    - `Set Locale... >> Country` . Change to `US (USA)`
    - Set the other fields (`Timezone`, `Keyboard` and `WiFi Country`) with your preference
6. Click in `Ok` to finish this setup. **Note:** Do not reboot yet!!!.

7. Connect the Raspberry to an WiFi clicking on its ícon on upper right corner of display
8. Click in the Raspberry ícon (see step 1), `Shutdown... >> Reboot` to finish the setup

Setup remote access - SSH (Secure Shell)

Make sure your Raspberry Pi is properly set up and connected. ou will need to note down the IP address of your Pi in order to connect to it later. Using the `ifconfig` command in a terminal will display information about the current network status, including the IP address, or you can use `hostname -I` to display the IP addresses associated with the device.

> Skip the steps bellow if the SSH is already enabled

Enable SSH

1. Launch **Raspberry Pi Configuration** from the **Preferences** menu
2. Navigate to the **Interfaces** tab
3. Select **Enabled** next to **SSH**
4. Click **OK**

Alternatively, `raspi-config` can be used in the terminal:

1. Enter `sudo raspi-config` in a terminal window
2. Select **Interfacing Options**
3. Navigate to and select **SSH**
4. Choose **Yes**
5. Select **Ok**
6. Choose **Finish**

SSH using Linux or Mac OS

You can use SSH to connect to your Raspberry Pi from a Linux computer, a Mac, or another Raspberry Pi, without installing additional software. You will need to know your Raspberry Pi's IP address to connect to it. To find this, type `hostname -I` from your Raspberry Pi terminal.

To connect to your Pi from a different computer, copy and paste the following command into the terminal window but replace `<IP>` with the IP address of the Raspberry Pi. Use `Ctrl + Shift + V` to paste in the terminal.

```
HOST:~$ ssh pi@<IP>
```

When the connection works you will see a security/authenticity warning. Type yes to continue. You will only see this warning the first time you connect.

```
The authenticity of host <IP> (<IP>) can't be established.
ECDSA key fingerprint is
SHA256:KKQF9QAUlNOF1okyQcqdRiTd9m0hck0/lcYpuJeCbqk.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added <IP> (ECDSA) to the list of known hosts.
```

In the event your Pi has taken the IP address of a device to which your computer has connected before (even if this was on another network), you may be given a warning and asked to clear the record from your list of known devices. Following this instruction and trying the `ssh` command again should be successful.

Next you will be prompted for the password for the pi login. You should now be able to see the Raspberry Pi prompt, which will be identical to the one found on the Raspberry Pi itself.

X-forwarding

You can also forward your X session over SSH, to allow the use of graphical applications, by using the `-Y` flag:

```
HOST:~$ ssh -Y pi@<RASPBERRY_IP>
```

Now you are on the command line as before, but you have the ability to open up graphical windows. For example, typing:

```
geany &
```

For detailed explanation, see SSH using Linux or Mac OS.

SSH Trick

Currently, you need to type your password each time you connect with the RPi. With the use of ssh-keys, we can automate this process.

1. Generate ssh-keys in the VM.

```
HOST:~$ cd ~/.ssh
HOST:~$ ssh-keygen -t rsa
  Generating public/private rsa key pair.
  Enter file in which to save the key (/home/<YOUR_USER>/.ssh/id_rsa):
doogie_mouse_rsa
  Enter passphrase (empty for no passphrase): <empty>
  Enter same passphrase again: <empty>
  Your identification has been saved in rpizero_rsa.
  Your public key has been saved in rpizero_rsa.pub.
  ...
```

> Optionally you can choose a different rsa-name (required if you are planning to use multie keys for different systems) and set a passphrase (increasing security). In my setup I left the passphrase empty (just hitting enter).

2. Set correct permisions of the key-set

```
HOST:~$ chmod 700 doogie_mouse_rsa doogie_mouse_rsa.pub
```

3. Send a copy of the public key to the RPi so it can verify the connection

```
cat ~/.ssh/doogie_mouse_rsa.pub | ssh pi@<RASPBERRY_IP> "mkdir -p ~/.ssh &&
cat >> ~/.ssh/authorized_keys"
```

4. Configure ssh connection in `ssh_config`

```
HOST:~$ sudo gedit /etc/ssh/ssh_config
```

Depending on the configuration of `dhcpcd.conf` on the RPi, add the following lines in the file end:

```
Host doogie-mouse
  HostName <RASPBERRY_IP>
  IdentityFile ~/.ssh/doogie_mouse_rsa
  User pi
  Port 22
```

5. Allow bash to invoke the configuration upon a ssh-call

```
HOST:~$ ssh-agent bash
HOST:~$ ssh-add ~/.ssh/doogie_mouse_rsa
```

6. Test connection:

```
HOST:~$ ssh -Y doogie-mouse
```

You should now be logged in onto the Raspberry Pi via SSH, without entering your password.

Installing ROS

Increase Swap Space

In order to ensure there is enough swap space to compile ROS, make the following modificationb below, then reboot the Raspberry Pi Zero so the change takes affect.

```
RPI:~$ sudo nano /etc/dphys-swapfile
```

and set the variable **CONF_SWAPSIZE=1024**.

Setup ROS Repositories

```
RPI:~$ sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu buster
main" > /etc/apt/sources.list.d/ros-latest.list'
RPI:~$ wget https://raw.githubusercontent.com/ros/rosdistro/master/ros.key
-O - | sudo apt-key add -
```

Now, make sure your Debian package index is up-to-date:

```
RPI:~$ sudo apt-get update
RPI:~$ sudo apt-get upgrade
```

Install Bootstrap Dependencies

```
RPI:~$ sudo apt-get install -y python-rosdep python-rosinstall-generator
python-wstool python-rosinstall build-essential cmake
```

Initializing rosdep

```
RPI:~$ sudo rosdep init
RPI:~$ rosdep update
```

Instalation

Now, we will download and build ROS Kinetic

Create a catkin Workspace

In order to build the core packages, you will need a catkin workspace. Create one now:

```
RPI:~$ mkdir -p ~/catkin_ws_isolated
RPI:~$ cd ~/catkin_ws_isolated
```

Next we will want to fetch the core packages so we can build them. We will use wstool for this. We will install ROS-Comm (ROS package, build, and communication libraries without GUI tools).

```
RPI:~$ rosinstall_generator ros_comm ros_control sensor_msgs
diff_drive_controller joint_state_controller control_toolbox --rosdistro
```

```
kinetic --deps --wet-only --exclude roslisp --tar > kinetic-ros_comm-
wet.rosinstall
RPI:~$ wstool init src kinetic-ros_comm-wet.rosinstall -j2
```

This will add all of the catkin or wet packages in the given variant and then fetch the sources into the `~/catkin_ws_isolated/src` directory. The command will take a few minutes to download all of the core ROS packages into the src folder. The -j2 option downloads 2 packages in parallel.

Resolve Dependencies

Before you can build your catkin workspace, you need to make sure that you have all the required dependencies. We use the `rosdep tool` for this, however, a couple of dependencies are not available in the repositories. They must be manually built first.

libconsole-bridge-dev:

```
RPI:~$ sudo apt-get install libconsole-bridge-dev
```

liblz4-dev:

```
RPI:~$ sudo apt-get install liblz4-dev
```

Resolving Dependencies with rosdep

The remaining dependencies should be resolved by running rosdep:

```
RPI:~$ cd ~/catkin_ws_isolated
RPI:~$ rosdep install -y --from-paths src --ignore-src --rosdistro kinetic
-r --os=debian:jessie
```

This will look at all of the packages in the src directory and find all of the dependencies they have. Then it will recursively install the dependencies.

The `--from-paths` option indicates we want to install the dependencies for an entire directory of packages, in this case src. The `--ignore-src` option indicates to rosdep that it shouldn't try to install any ROS packages in the src folder from the package manager, we don't need it to since we are building them ourselves. The `--rosdistro` option is required because we don't have a ROS environment setup yet, so we have to indicate to rosdep what version of ROS we are building for. Finally, the `-y` option indicates to rosdep that we don't want to be bothered by too many prompts from the package manager.

After a while rosdep will finish installing system dependencies and you can continue.

Building the catkin Workspace

Once you have completed downloading the packages and have resolved the dependencies, you are ready to build the catkin packages.

Invoke `catkin_make_isolated`:

```
RPI:~$ sudo ./src/catkin/bin/catkin_make_isolated --install -
DCMAKE_BUILD_TYPE=Release --install-space /opt/ros/kinetic -j1
```

Source the setup.bash in the ~/.bashrc, so that ROS environment variables are automatically added to your bash session every time a new shell is launched:

```
RPI:~$ echo "source /opt/ros/kinetic/setup.bash" >> ~/.bashrc
```

Maintaining a Source Checkout

Adding Released Packages

You may add additional packages to the installed ros workspace that have been released into the ros ecosystem. First, a new rosinstall file must be created including the new packages (Note, this can also be done at the initial install). For example, if we have installed `ros_comm`, `sensor_msgs` and `ros_control`, but want to add `roscpp_tutorials`, the command would be:

```
RPI:~$ cd ~/catkin_ws_isolated
RPI:~$ rosinstall_generator ros_comm ros_control sensor_msgs
roscpp_tutorials --rosdistro kinetic --deps --wet-only --tar > kinetic-
custom_ros.rosinstall
```

You may keep listing as many ROS packages as you'd like separated by spaces.

Next, update the workspace with wstool:

```
RPI:~$ wstool merge -t src kinetic-custom_ros.rosinstall
RPI:~$ wstool update -t src
```

After updating the workspace, you may want to run rosdep to install any new dependencies that are required:

```
RPI:~$ rosdep install -y --from-paths src --ignore-src --rosdistro kinetic
-r --os=debian:jessie
```

Finally, now that the workspace is up to date and dependencies are satisfied, rebuild the workspace:

```
RPI:~$ sudo ./src/catkin/bin/catkin_make_isolated --install -
DCMAKE_BUILD_TYPE=Release --install-space /opt/ros/kinetic
```

ROS Hello World!

To check if ROS is working properly, create a new workspace where the Doogie Mouse packages will be installed and built:

```
RPI:~$ mkdir -p ~/doogie_ws/src
RPI:~$ cd ~/doogie_ws
RPI:~$ catkin_make
```

Now, clone the `doogie_welcome` package into the workspace and build it:

```
RPI:~$ cd ~/doogie_ws/src
RPI:~$ git clone https://github.com/doogie-mouse/doogie_welcome.git
RPI:~$ cd ~/doogie_ws/
RPI:~$ catkin_make
```

Run the nodes:

```
RPI:~$ source ~/doogie_ws/devel/setup.bash
RPI:~$ roslaunch doogie_welcome doogie_welcome.launch
```

> **Note:** Source the devel setup.bash in the ~/.bashrc, so that ROS environment variables are automatically added to your bash session every time a new shell is launched:
>
> ```
> RPI:~$ echo "source ~/doogie_ws/devel/setup.bash" >> ~/.bashrc
> ```

In another terminal of your machine, go inside the Doogie Mouse Raspberry again using SSH and do the next steps.

You could see all the topics created by running:

```
RPI:~$ rostopic list
```

This command will reproduce a list like

```
pi@doogie-mouse:~ $ rostopic list
/cpp_chatter
/python_chatter
/rosout
/rosout_agg
```

To see what is happening in the topic `cpp_chatter`, run the command:

```
RPI:~$ rostopic echo /cpp_chatter
```

Usage tools

The tools listed below are highly recommended.

Installing terminal Terminator

Run the command:

```
RPI:~$ sudo apt-get install terminator
```

See the Terminator's documentation for detailed functionalities description.

Installing htop

To see CPU and RAM memory usage by Raspberry, install htop

```
RPI:~$ sudo apt-get install htop
```

References

This software setup was made based on:

- Installing ROS Kinetic on the Raspberry Pi
- Installing ROS Indigo on the Raspberry Pi
- SSH-Keys : pi-user