Chapter 1.

The smart boombox cookbook

This document describes how to create a *boombox* that sounds great and is smart enough that you can ask it to play the music you want to hear.

A Raspberry Pi 4 running GNU/Linux drives the smarts and a Digital to Audio Converter (DAC), amplifier and quality speakers create the great sound.

Also described is a *battery boombox* if you want a portable boombox with batteries, but without the computer components.

This document has the following sections that should be followed if you're building either the smart or the battery boombox:

If you are building a *smart-boombox*, also follow these sections:

Once your system is built, installed and configured, whether smart or battery, follow the final section.

1.1 Overview

Wikipedia defines a *boombox* as:

> ...a transistorized portable music player featuring one or two cassette tape recorder/players and AM/FM radio, generally with a carrying handle. Beginning in the mid 1980s, a CD player was often included.
>
> https://en.wikipedia.org/wiki/Boombox

Today, cassettes are ancient history, CDs are rare and AM/FM is used less frequently, but the concept of a boombox lives on. There are other new sources of music that a modern boombox can leverage, including:

- Bluetooth connection to a device that can play music such as a cell phone,
- USB drives with song files,
- "Aux in" jack to drive music from other devices, such as a CD player or phonograph, and
- Internet streaming of radio stations, music playing apps and services such as Pandora and Spotify.

The resulting device can be considered either a boombox that is really smart, or as a general purpose computer that sounds great.

The boombox has a *2.1 channel* amplifier. That means the amp has outputs for two full range speakers and one subwoofer that drives the lower frequencies. It has a down-firing subwoofer which helps to keep gravity on its side. The

There is also a headphone amplifier, on the smart boombox, if you want to listen to music and keep it quiet.

There are covers for the left and right speakers, and for the subwoofer to protect them. There are no covers (though you could add them) for the two passive radiators. The philosophy is to protect the expensive components, but to leave the less expensive passive radiators exposed for a better look and sound.

1.1.1 A battery boombox

If you want portablity, but not the smarts, all of the computer components can be omitted but batteries and a controller are added. If you build or buy one of these, you won't be able to:

- Give it voice commands,
- Stream music from the Internet,
- Listen to music through headphones, or
- Have access to a general purpose computer

This approach reduces the cost and the complexity of the project. You could also first build a basic boombox, then add the smarts later.

1.2 Choices made

This section describes the choices made in the design of the boombox, possible alternatives, and why they were made. The following table moves from the low-level hardware to the high-level software.

Table 1-1 Choices made and alternatives

Component	Choice made	Alternatives	Reasoning
Boombox frame	Baltic birch + solid moldings - 3 speaker + 1 top enclosure	3D-printed plastic, MDF, ammo boxes, etc.	Many builders believe Baltic Birch is *the* best material for speaker enclosures. Wood is more recyclable and more aesthetically pleasing than plastic.
Amplifier type	2.1 channel amp	2.0 channel amp, DAC HAT amp	A 2.1 amp does the crossover from highs and lows. A 2.0 amp would require dedicated crossover components or no subwoofer which would lose the lows. A DAC HAT amp has no tone/volume controls nor the power needed to drive the speakers.
Left/Right speaker enclosures	Ported	Sealed	The ported enclosures just sound better than sealded. Almost everyone agrees.
Subwoofer enclosure	Sealed + passive radiators	Sealed or ported w/o passive radiators	Sealed enclosures with passive radiators allow a lower frequency than the other alternatives - the lower the bass, the better!
Smart or basic boombox	Smart, but you can choose basic	Basic	If you want a voice assistant with modern streaming options, you should go with the smart boombox. If not, then buy or build the basic boombox. Both are described.
System On a Chip (SOC)	Raspberry Pi 4	Pine 64, Jetson Nano, others	The Raspberry Pi 4 with 4 GB of memory seems to have the widest acceptance and support, and is capable of driving a general purpose computer. Ideally there would be an SoC that runs totally free software, but none could be found.
Operating system	GNU/Linux Ubuntu Desktop	Raspbian, Pycroft, Volumio, etc.	Ubuntu Desktop GNU/Linux was chosen very closely over Raspbian (aka Raspberry Pi OS). There are other OS's dedicated to one function such as Pycroft and Volumio. If you have the horsepower to drive a general purpose computer, why not use it?
Desktop environment	GNOME shell	KDE, Mate, many others	The default desktop that ships with Ubuntu is GNOME
Voice software	Mycroft	Rhasspy, many others	Mycroft seems to be the most widely accepted, free and open choice available with an important focus on privacy
Music playing software	Emby	Jellyfin, others	Not sure - Jellyfin may be a better choice
Low level audio	pulseaudio, alsa	None	As Mycroft and Emby require pulseaudio, which requires ALSA, there is no other option. The code gets pulled in automatically.

1.3 Bill of materials

This section describes the hardware and software you will need to build such a boombox.

The options, in the order of least to most expensive, are:

1. Build everything yourself - just figure out how to build the frame,
2. Buy an unassembled frame, glue it together, then buy and install all other parts,

   (**TODO: smartboomboxes.com model 100**)
3. Buy an assembled frame, then buy and install all other parts,

   (**TODO: smartboomboxes.com model 200**)
4. Buy a fully-assembled basic boombox, or

   (**TODO: smartboomboxes.com model 900 or 1100**)
5. Buy a fully-assembled smart boombox.

   (**TODO: smartboomboxes.com model 1000 or 1200**)

1.3.1 Hardware for the basic box

Description	Qty	Cost	Source
Wooden frame (unassembled)	0/1	$69?	At smartboomboxes.com **SOON**?
Wooden frame (assembled)	0/1	139?	At smartboomboxes.com **SOON**?
2.1 amp 50/50/100W w/bluetooth	1	$28	www.amazon.com/gp/product/B09HKB44JQ
12-24v Power supply	1	$20?	??? **TODO**: test different power supplies
Tang Band 3" full range speakers	2	$66	PE: 264-911 $66 (or $57 for 4 or more)
Peerless passive radiators	2	$16	PE 264-1060
6" subwoofer speaker grill	1	$3	PE: 260-371
4" Left and Right speaker grill	2	$3	PE 269-2220
Lighted rocker switch	1	1	www.amazon.com/gp/product/B07TH9YB5H/ 10 for $8 (included with frame above)
Feet - 1-1/4" - all rubber or copper tubing and short rubber	4	$10	www.amazon.com/gp/product/B00S2D3FPY -OR- back to 1" copper with 1/4" rubber feet
Handle	1	$4	PE: 262-314
Subtotal		$231	unassembled box, or $321 assembled

1.3.2 Additional smart boombox hardware

Description	Qty	Cost	Source
Raspberry Pi 4 w/4 GB memory, plus power supply and heat sinks	1	$70	www.canakit.com/raspberry-pi-4-basic-kit.html **NOTE: no longer available :((** due to global chip shortage
RasPi 5v Power supply	1	0	Comes with above kit
RasPi Heat sinks	1	0	Comes with above kit
RasPi inline switch	1	$7	www.canakit.com/raspberry-pi-4-on-off-power-switch.html
64 GB Micro SD card/holder	1	$13	https://tinyurl.com/ymat2es4
"DAC HAT" HiFiBerry DAC+	1	$45	www.hifiberry.com/shop/boards/hifiberry-dac2-pro/
3.5mm 2 female to 1 male cable (Y cable)	1	$1	PE 240-1026
1' 3.5mm male to female	2	$6	https://www.amazon.com/gp/product/B087CLK6Q5?th=1
RCA female to 1 3.5mm male	1	$6	https://www.amazon.com/gp/product/B000I963XQ
USB microphone	1	$14	www.amazon.com/gp/product/B0779PKLV9/ref=ppx_yo_dt_b_asin_title_o07_s00
Electrical box, outlet and cover	1	$6-8	Hardware store
USB drive with music files	1	?	Optional
Subtotal		$158	

```
Assembled frame 139+34+20+57+16+3+3+5+10+4= $291
Unassembled     69+34+20+57+16+3+3+5+10+4= $221
Computer parts  70+7+13+45+1+2+14+6= %158

Tang Band W5-1138SMF 5-1/4" Subwoofer 264-917 $49.98
6" Grill 260-371 $3.19
LBB-5Sv2 5 x 18650 Lithium Battery Charger Board/Module 325-210 23.49
Skar 1: tweeters -
https://www.amazon.com/Skar-Audio-TWS-01-Neodymium-Tweeters/dp/B01HQRAMTS
$23.99
https://www.amazon.com/gp/product/B087CLK6Q5
```

Most sources are either from Parts Express (partsexpress.com - abbreviated with "PE: <part-number>") or Amazon or other sources where the URL is provided.

So the cost would range from approximately $221 (unassembled box and no computer) to $349 (assembled box with computer).

Of course, the product sources and prices will fluctuate.

1.3.3 Equipment for first time set up

To set up the system the first time, you will also need the following.

- A keyboard, mouse and monitor
- HDMI cable and possibly a HDMI to micro-HDMI converter
- Wi-Fi Internet or wired Ethernet
- A PC or other device with an SD card port
- A device with an "aux cable" (3.5mm male connector), such as a cell phone, that can play music to test the amplifier.

1.3.4 Software

You will need the following software:

- A Linux distribution - Ubuntu 22.04 Desktop, also known as *Jammy Jellyfish*, is recommended and is what is documented in this guide.

  Raspberry Pi OS (aka Raspbian) is an alternative but is not documented.
- If you have a Windows computer, the Win-32 Disk imager is recommended for copying the Linux image to the micro SD card.
- If you have a Linux computer, the `dd` command can copy the Linux image.

1.4 Assemble the frame

The entire boombox consists of four separate enclosures:

Left channel	Bottom left. This enclosure is ported with a 1” diameter, 3” long tube.
Subwoofer	Bottom center. This enclosure is sealed with two passive radiators facing forward and backward. These are elements designed to move sympathetically with the energy of the subwoofer. Their extra mass allows for a resonant frequency lower than otherwise possible. The subwoofer is facing down.
Right channel	Bottom right. This enclosure is also ported.
Components	Top. This enclosure has two sliding panels so that it can be completely open and give maximum access to the computer and electronic components.

If you bought the unassembled kit, you can watch a video **TODO-create-video1** on how to assemble it.

1.5 Install the speakers and amplifier

Now that the frame is assembled, you can install the speakers, ports, passive radiators, amplifier and wiring.

You can watch a video **TODO-create-video2** on how to install them.

1.5.1 Test amplifier and speakers

Now is time to test that your amplifier and speakers are working, before getting to the computer parts. To do so, perform the following steps.

- If you're building a smart boombox, unplug the output of the 5V power supply from the in-line switch to the RasPi in the top compartment (**TODO**: get a picture).
- Set the on/off switch on the front left to **off**.
- Plug the main power cord in to 115v outlet.
- Set the on/off switch on the front left to **on**. The light in the switch should come on.
- Connect a device with an "aux cord" to "Aux-in" (**TODO**: get a picture), and play some music!

If you don't hear any music (**TODO**: add some words on debugging).

1.6 Load GNU/Linux on the micro-SD card

If you are building the basic boombox, you can skip this entire section.

Before starting with the hardware, you need a GNU/Linux image copied onto a micro SD card. This card will be plugged into your RasPi 4 as the operating system that will drive your boombox.

This section is comprised of the following sub-sections:

1.6.1 Prepare to copy an SD card

The computer you use to write to the micro-SD card must have a port for a card.

Prepare on Linux

If you have a Linux system with an SD card reader, you can use the rpi-imager tool. To do so, perform the following tasks.

- Put a micro-SD card into an SD adapter.
- Plug the SD adapter into the card reader.
- Download and install rpi-imager.

```
$ sudo apt-get install -y rpi-imager
...
```

- Run the tool

```
$ rpi-imager
```

- You should see a window as shown in the following figure.
  - Select the operating system. In this example it is Ubuntu Desktop 22.04 LTS.

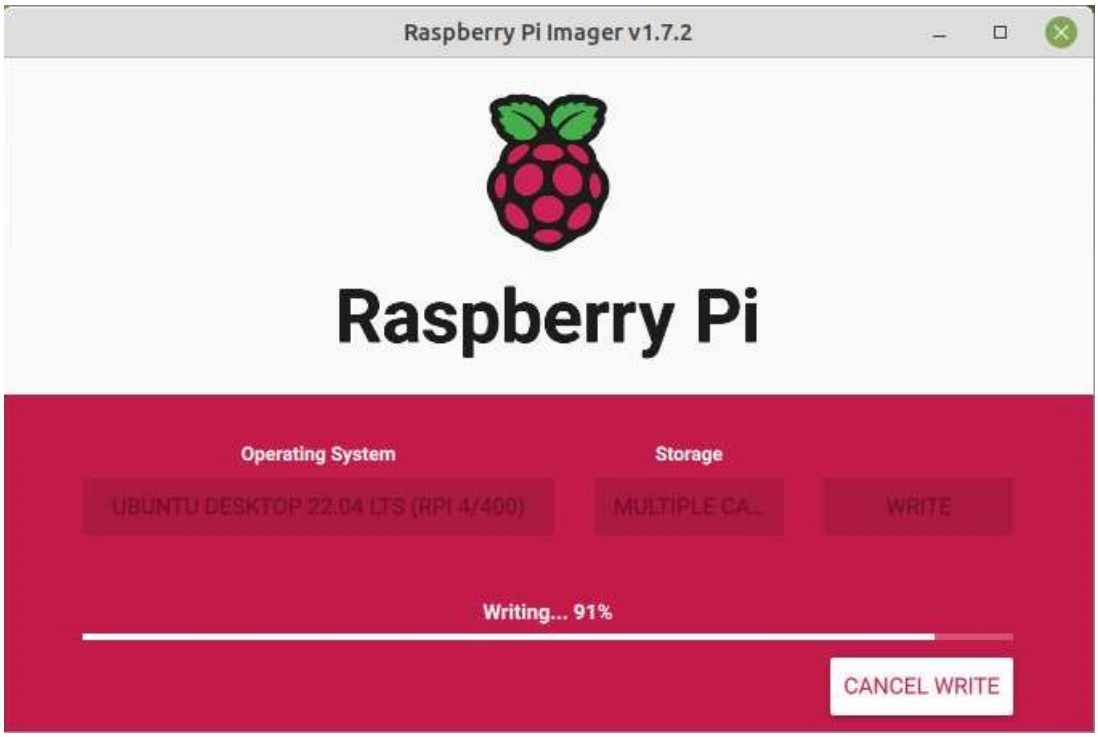

Figure 1-1 rpi-imager main window

Prepare on Windows

If you only have access to a Windows system Install the Win 32 disk imager

This software will allow you to copy a downloaded Linux image onto a micro-SD card. If you have an application that will do this, you can skip this step.

- Download the Win 32 disk imager to a Windows PC

  `https://sourceforge.net/projects/win32diskimager/`

- Install it.

1.6.2 Download Ubuntu Desktop

To download Ubuntu, perform the following steps:

- Download an Ubuntu Desktop image from:

  `https://ubuntu.com/download/raspberry-pi`

- Click the Green **Download 64-bit** button on the left side of the page. This version is recommended because it includes "Long Term Service" (until 2025).

  TODO: get screen shots

- A .zip or .xz file should be downloaded to your PC. For example:

  `ubuntu-20-04.3-preinstalled-server-arm64++raspi.img.xz`

  The `.xz` suffix implies the file is compressed in a format other than zip.

- Unzip the file with a zip tool. Most Operating Systems have these standard today. When the file is unzipped, it should have a name similar to:

```
ubuntu-20-04.3-preinstalled-server-arm64++raspi.img
```

You should now be ready to copy the image to a micro-SD card.

1.6.3 Copy Ubuntu image to micro-SD card

If you have a brand new SD card, you usually don't need to format it. Most come preformatted with the FAT32 file system.

- **TODO**: describe Win32installer with some screen shots
- **TODO**: Describe Raspberry Pi Imager on Linux => easier to use.

At this point you should have a micro-SD card with Ubuntu Desktop on it which should boot the RasPi.

1.7 Connect the computer hardware

To connect all the computer hardware, perform the following steps:

TODO: add pictures

Robert: start here! - the micro-SD is plugged in and you should not need a wired Internet. Plug the USB drive with music in the front.

- Plug the micro-SD card into the back of the RasPi.
- You can access the Internet using either Wi-Fi or with an Ethernet patch cord with an RJ-45.
- Plug the 5v power supply with a USB-C end into the RasPi 4.

For the initial set up, a keyboard, monitor and mouse are needed. Ideally there will be a way of setting up "headlessly", but that's not available yet.

- Connect the mouse and keyboard to the USB connections on the RasPi.
- Connect the monitor to the RasPi with the HDMI cable.

1.7.1 Boot the RasPi

To prepare to boot the Raspberry Pi 4 for the first time, perform the following steps.

- Slide both the top and back covers off of the frame to have full access to the hardware.
- If you haven't already, download 64-bit Ubuntu 22.04 (LTS) - described in "Download Ubuntu Desktop" on page 10.
- If you haven't already, copy Linux to the micro-SD card and plug it in the back of the RasPi. This is described in "Copy Ubuntu image to micro-SD card" on page 11
- Plug monitor, keyboard, mouse and microphone into the Raspberry Pi. The Raspberry Pi 4 has two micro-HDMI jacks as it can drive two displays. If you have only one display, use the jack closest to the power supply jack (the one to the left).
- Plug in boombox main power. The small inline button switch inside powers the Raspberry Pi. (**TODO**: screen shot)

First boot

When you supply power to the Raspberry Pi, it should start booting.

- If the RasPi switch has a red LED below the on/off button. If it is not glowing red, push the on/off button once.
- Observe the top, back, left of the RasPi (which is under the DAC Hat). There are two LEDs:
  - The LED to the left should glow solid red. This signifies the RasPi has 5V DC power.
  - The LED to the right of the red one should flicker green. This signifies that an operating system is communicating with the CPU. If there is a red light, but no green one, this probably means that the micro-SD card does not have Linux properly installed.

Important: Never turn the RasPi off without first shutting Linux down with the `halt` command. Doing so can damage the operating system and possibly even the RasPi itself.

- Observe the monitor. You should see a rainbow colored splash screen, then the Ubuntu desktop should initialize.

Initial configuration

- A welcome screen should pop up.
- On the *Welcome* window, choose your language and click **Continue**.
- On the *Keyboard layout* window, choose your keyboard layout and click **Continue**.
- On the *Wireless* window, configure a Wi-Fi network unless you plan to only use a hard-wired ethernet cable.
- On the *Where are you?* window, choose your time zone.
- On the *Who are you?* window, set the following values.
  - Set your name.
  - Choose your computer's name (host name).
  - For a user name and password `pi` is recommended as it is documented in the reminder of this document.
  - For the last option, **Log in automatically** is recommended.
  - Click **Continue**. The install process will take a number of minutes configuring and will reboot itself.
- An *Online Accounts* window should appear. Click **Skip**.
- Choose an option on the *Livepatch* window.
- Choose an option on the *Help Improve Ubuntu* window and click **Next**.
- Click **Next** at the *Privacy* window.
- Click **Done** at the *Ready to go* window.

Ubuntu Desktop should now be installed.

1.8 Install and configure software

To configure Ubuntu, perform the following sections.

Install SSH server and other software

The ssh server is not installed by default on Ubuntu desktop. It is recommended that you install it so you can access your boombox remotely.

- Open a terminal session by right-clicking the mouse anywhere on the desktop and choosing **Open in Terminal**. You should see a window pop up.
- Install open SSH server, and other packages with the following command.

```
$ sudo apt-get install -y openssh-server
[sudo] password for pi:
...
```

- After it installs sshd should be running. Verify with the following command:

```
$ service sshd status
...active (running)
```

- You should have either a Wi-Fi (`wlan0`) or a hard-wired (`eth0`) connection. To verify, enter the following command. Note your IP address.

```
ip a
1: lo:
...
2: eth0:
...
inet 192.168.1.229
3: wlan0:
...
inet 192.168.1.x
```

  In this example, the IP address is **`192.168.1.229`**.

Start an SSH session

You should now be able to start an SSH session as the user `pi`, if you want to continue from another desktop system. You can use **`putty`** to SSH in from a Windows PC, or just use the **`ssh`** command from a Linux or macOS terminal session.

Upgrade your system

- To prevent your system doing upgrades without your knowledge, issue the following command. Use the Tab key to move the cursor to **No** and press Enter.

```
$ sudo dpkg-reconfigure unattended-upgrades
Applying updates on a frequent basis is an important part of keeping
systems secure. By default, updates need to be applied manually using
package management tools....

Automatically download and install stable updates?
       <Yes>                                <No>
```

- *Update* the system which prepares for the latest code for all installed packages.

```
$ sudo apt-get update
```

- *Upgrade* your system so you have all the latest code. This step could take 10 to 20 minutes.

```
$ sudo apt-get upgrade
...
After this operation, 38.3 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
...
```

- If you are prompted to restart services, use the Tab key to move to **Ok** and accept all defaults.

Your system should now be at the latest software levels.

Install other software

- Install other software with the following command:

```
$ sudo apt-get install -y vim git mlocate net-tools python3
[sudo] password for pi:
...
```

  - `vim` is the classic UNIX/Linux *full-screen line* editor.
  - `git` allows you to copy and manipulate code from github.
  - `mlocate` provides the **`locate`** and **`updatedb`** commands to find files and directories extremely fast.
  - `net-tools` provides classic networking commands such as **`ifconfig`**.
  - `python3` provides an environment for the popular Python language.
  - `pulseaudio` drives the speakers and microphones.

Continue Linux configuration

The following steps are recommended to configure your Linux system.

- This step changes the default editor to **`vim`**. If you are happy with the **`nano`** editor, you can skip it. Enter the following command and choose **3** for vim.

```
$ sudo update-alternatives --config editor
[sudo] password for pi:
There are 4 choices for the alternative editor (providing
/usr/bin/editor).

  Selection    Path                Priority   Status
------------------------------------------------------------
* 0            /bin/nano            40        auto mode
  1            /bin/ed             -100       manual mode
  2            /bin/nano            40        manual mode
  3            /usr/bin/vim.basic   30         manual mode
  4            /usr/bin/vim.tiny    15         manual mode

Press <enter> to keep the current choice[*], or selec a number: 3
...
```

- Change the vim color scheme because dark blue comments are difficult to read. Add the line **`colorscheme desert`** line as shown below.

```
$ sudo vim /etc/vim/vimrc
...
if has("syntax")
  syntax on
  colorscheme desert
endif
...
```

- Don't require passwords on sudo commands by adding **NOPASSWD:** to the group sudo.

```
$ sudo visudo
...
# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) NOPASSWD: ALL
...
```

- Turn off the annoying autoindent in vim.

```
$ cd /usr/share/vim/vim82
$ sudo mv indent.vim indent.vim.dontuse

Note: if the above doesn't work, try these
$ cd /usr/share/vim/vim90
$ sudo mv sh.vim sh.vim.dontuse
$ mkdir -p ~/.vim/indent
$ echo "let b:did_indent = 1" > ~/.vim/indent/html.vim
```

Update the user pi

The user and group named `pi` should have been created as part of the installation process.

- Verify that the pi user and group exist with the following commands.

```
$ id pi
uid=1000(pi) gid=1000(pi) groups=1000(pi),4(adm),24(cdrom),27(sudo),
30(dip),46(plugdev),123(lpadmin),135(lxd),136(sambashare)
$ getent group pi
pi:x:1000:
```

- Add the groups `audio` and `video` to the `pi` user.

```
$ sudo usermod -a -G audio,video pi
$ id pi
uid=1000(pi) gid=1000(pi)
groups=1000(pi),4(adm),24(cdrom),27(sudo),29(audio),30(dip),44(video),46(
plugdev),123(lpadmin),135(lxd),136(sambashare)
```

In the sections that follow, ***all commands should be run as pi***. If you "su to root" strange things will happen

Continue configuration

- Create a file name `.bash_profile` in pi's home directory. It is run when a shell is started. Because the directory is owned by `pi`, you should not need to precede **vi** with **sudo**.

```
# vi .bash_profile
alias la='ls -latr'
alias vi=/usr/bin/vim
export CPS="/home/pi/mycroft-core/mycroft/skills"             # common
play skill
export DNK="/home/pi/mycroft-dinkum/skills/play-music.mark2"  # dinkum
music skill
export JZ2="/home/pi/mympdplaylist-skill"
export LOG="/var/log/mycroft"
export MCT="/home/pi/mycroft-tools/usr/local/sbin"
export MPC="/opt/mycroft/skills/mpc-skill-mike99mac"
export
PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/home/
pi/mycroft-core/bin"
```

```
export SKI="/opt/mycroft/skills"
export SYS="/etc/systemd/system"
export Z="/usr/local/sbin"
```

- The alias **la** lists files with details in reverse chronological order. That means the most recently modified files are shown at the bottom.
- The alias **vi** uses the **vim** editor not the original **vi**.
- The environment variable **LOG** is a shortcut to the Mycroft log files directory.
- The environment variable **PATH** includes the Mycroft executables in the executables search path.
- The environment variable **SKI** is a shortcut to the Mycroft skills directory.
- The environment variable **SYS** is a shortcut to the systemd directory where service configuration files are stored.

Install other useful tools

The author of this guide, `@mike99mac`, has also written some small tools to help with the installation and testing of Mycroft and associated audio resources.

- Clone the `mycroft-tools` package in pi's home directory with the following commands.

```
$ cd
$ git clone https://github.com/mike99mac/mycroft-tools.git
...
```

- Change to the newly installed directory and run the setup script.

```
$ cd mycroft-tools
$ sudo ./setup.sh
Copying all scripts to /usr/local/sbin ...
Success!  There are new scripts in your /usr/local/sbin/ directory
```

- Show the files that were copied to `/usr/local/sbin/`.

```
$ ls/usr/local/sbin
gr     lsintent        lspairing  lsskills  lsvocab   testplay
lsenv  lsmycrofttools  lsskill    lstemp    setup.sh  testrecord
```

  The scripts **lsintent**, **lsskills** and **lsvocab** list the Mycroft intents, skills and vocabularies installed on your system.

  The scripts **testrecord** and **testplay** allow you to test your microphone and speakers. They will be used later.

  The script **lstemp** shows you the current temperature of your Raspberry Pi.

Turn off default sound card

Tell Ubuntu to load the HiFiBerry DAC HAT and to turn off the default Broadcom sound card by performing the following steps.

- Run the command **aplay -l**. This shows the hardware devices that can play sounds.
  - The `bcm2835` value is the Broadcom chip on the Raspberry Pi.
  - The `sndrpihifiberry` value is the Hifiberry DAC Pro+, so it has already been detected

```
$ aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: Headphones [bcm2835 Headphones], device 0: bcm2835 Headphones
[bcm2835 Headphones]
```

```
  Subdevices: 8/8
  Subdevice #0: subdevice #0
  ...
  Subdevice #7: subdevice #7
card 2: sndrpihifiberry [snd_rpi_hifiberry_dacplus], device 0: HiFiBerry
DAC+ Pro HiFi pcm512x-hifi-0 [HiFiBerry DAC+ Pro HiFi pcm512x-hifi-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
card 3: vc4hdmi0 [vc4-hdmi-0], device 0: MAI PCM i2s-hifi-0 [MAI PCM
i2s-hifi-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

- Change directory to `/boot/firmware/` and make a copy of the `config.txt` file.

```
$ cd /boot/firmware
$ sudo cp config.txt config.txt.orig
```

- Edit the configuration file and add the lines shown in bold.
  - The value `dtparam=audio=off` will turn off the audio on the broadcom chip, as it is of poor quality.
  - If you have an overscan problem, comment out the `disable_overscan` line.
  - Note there are `dtoverlay` values for both the HiFiBerry and Inno-maker DAC hats. Choose the setting corresponding to the DAC Hat you will be using.

```
$ sudo vi config.txt
[all]
kernel=vmlinuz
cmdline=cmdline.txt
initramfs initrd.img followkernel

[pi4]
max_framebuffers=2
arm_boost=1

[all]
# Enable audio output, I2C and SPI interfaces on GPIO header. As these
# parameters related to the base device-tree they must appear *before*
# other dtoverlay= specification
# Disable the built-in Broadcom sound card
dtparam=audio=off
dtparam=i2c_arm=on
dtparam=spi=on

# Comment out the following line if edges of the desktop appear outside
# edges of your display
# disable_overscan=1
...
#
# Other possible values
# Enable the HiFiBerry DAC HAT
# dtoverlay=hifiberry-dacplus,24db_digital_gain
# Enable the inno-maker DAC HAT
# dtoverlay=allo-boss-dac-pcm512x-audio,24db_digital_gain
# TODO: try these values recommended on:
# archimago.blogspot.com/2017/01/measurements-raspberry-pi-3-as-usb.html
# and see if there is a noticable difference in the sound
#arm_freq=800
#sdram_freq=400
#core_freq=400
```

```
#gpu_freq=300
#over_voltage=-4
#over_voltage_sdram=-4
#gpu_mem=16
```

- Reboot to test these changes.

```
$ sudo reboot
```

- When the system comes back up, start an SSH session as `pi`.
- List the devices again with the **`aplay -l`** command. Now the Broadcom device should be absent, and the hifiberry device is shown.

```
$ aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: sndrpihifiberry [snd_rpi_hifiberry_dacplus], device 0: HiFiBerry
DAC+ Pro HiFi pcm512x-hifi-0 [HiFiBerry DAC+ Pro HiFi pcm512x-hifi-0]
...
```

Test microphone and speakers

It is important to know your microphone and speakers are working. The scripts **`testrecord`** and **`testplay`** previously installed utilize **`arecord`** and **`aplay`** to save and play audio to and from a file.

- To test your microphone, issue the following command then speak into the microphone for up to five seconds. If the recording is successful the script will attempt to play it back.

```
$ testrecord
Testing your microphone for 5 seconds - SAY SOMETHING!
INFO: running command: arecord -r 44100  -f S24_LE -d 5 /tmp/test-mic.wav
Recording WAVE '/tmp/test-mic.wav' : Signed 24 bit Little Endian, Rate
44100 Hz, Mono
```

- Assuming that worked successfully, issue the following command to play back the recording.

```
$ testplay
Playing WAVE '/tmp/test-mic.wav' : Signed 16 bit Little Endian, Rate
16000 Hz, Mono
```

If you hear your voice played back, then your microphone and speakers are working.

If not, look at more options to the **`aplay`** and **`arecord`** commands.

TODO: add a section on preferences:

```
https://home.mycroft.ai/devices/preferences
```

Install Mycroft

To install Mycroft, perform the following steps.

- Change to the home directory and get the Mycroft code from github.com/

```
$ cd
$ git clone https://github.com/MycroftAI/mycroft-core.git
Cloning into 'mycroft-core'...
remote: Enumerating objects: 553610, done.
remote: Counting objects: 100% (1587/1587), done.
remote: Compressing objects: 100% (830/830), done.
...
```

- Run the `dev_setup.sh` script in the `mycroft-core/` directory.

```
$ cd mycroft-core
$ bash dev_setup.sh
```

... answer **Yes** to all questions ...

This step can 20+ minutes.

Set Mycroft to start at boot time

To set Mycroft to start when the Raspberry Pi boots, perform the following steps.

- Change to the systemd configuration directory.

```
$ cd $SYS
$ pwd
/etc/systemd/system
```

- Copy the file `mycroft.service` which should be in your **`mycroft-tools/`** directory. This will start Mycroft at boot time.

```
$ cp ~/mycroft-tools/mycroft.service .
$ cat mycroft.service
[Unit]
Description=Mycroft AI
After=pulseaudio.service

[Service]
User=pi
WorkingDirectory=/home/pi/
ExecStart=/home/pi/mycroft-core/bin/mycroft-start all
ExecStop=/home/pi/mycroft-core/bin/mycroft-stop
Type=forking
Restart=no
Environment="XDG_RUNTIME_DIR=/run/user/1000"

[Install]
WantedBy=multi-user.target
```

- Enable the service to start at boot time.

  Robert: skip the next two steps as you want to do development with mycroft and need the debug/console environment. Rather, get a wide SuperPuTTY window and enter **`mycroft-start debug`**.

```
# sudo systemctl enable mycroft.service
Created symlink
/etc/systemd/system/multi-user.target.wants/mycroft.service ...
```

- Reboot the system to test these changes.

```
$ sudo reboot
```

- When the system comes back up, ideally Mycroft will start talking to you. If it does, it will tell you the six digit pairing code. Write it down.

  Robert: skip this step and do the next one

```
I'm connected to the internet and need to be activated.
Open your browser and visit account dot mycroft dot A I slash pair
(https://account.mycroft.ai/pair) to register this device.
```

  If Mycroft is not speaking to you, check the log files in `/var/log/mycroft/`.

- You can also get the pairing code with the **`lspairing`** script from mycroft-tools. For example:

```
$ lspairing
Enter code at: https://account.mycroft.ai/pair
```

```
Pairing code : ABC123
Generated    : 2022-07-21 08:16:04
```

You should hear Mycroft say “Your device is now paired”.

Register your device with Mycroft

You must register your device with Mycroft. To do so, perform the following steps.

- Point a browser to `https://account.mycroft.ai/pair`
- You should see a page similar to the following figure. Supply the pairing code and a description of the device and its location.

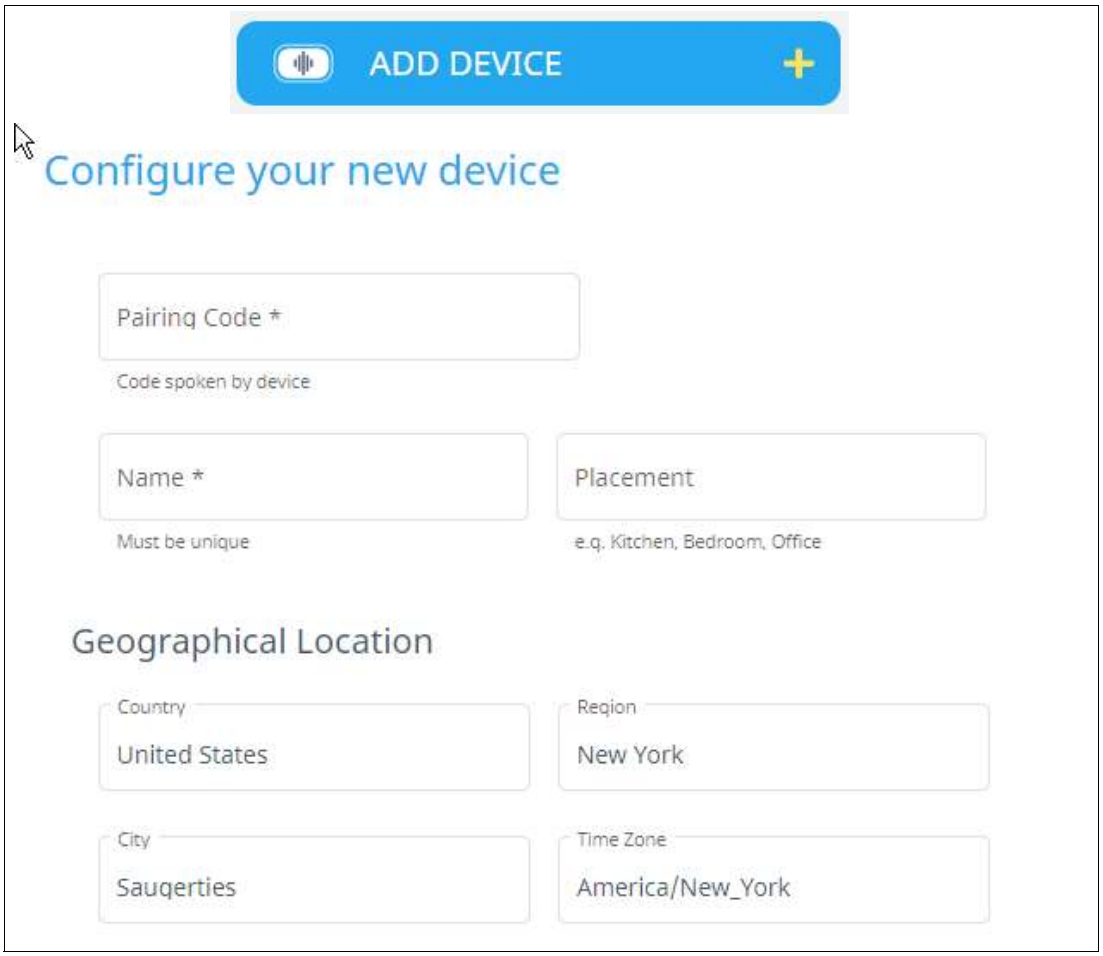

Figure 1-2 Adding a device to the Mycroft registry

- Enter the pairing Code, the name of your device and optionally other information.
- Click **Next**.
- You should see a page with the heading *Your Device is Ready*. Click **Finished**.
- Very soon after you should hear Mycroft speak:

  “Pairing is now complete, please wait a few moments while skills are loading.”

- Then:

```
"Now I am ready for use" .
Try asking me things like "hey mycroft, what's the weather", or
"hey mycroft, play the news"
...
```

- Try asking Mycroft some questions.
- Mycroft should now be running. Verify with the **`lsenv`** command that is part of the mycroft-tools repo:

```
$ lsenv
Status of mycroft:
 -) WARNING: mycroft is not running as a service ... checking for
processes ...
    WARNING: no processes matching mycroft found
------------------------------------------------------------------
Status of emby-server:
Unit emby-server.service could not be found.
 -) WARNING: emby-server is not running as a service ... checking for
processes ...
    WARNING: no processes matching emby-server found
------------------------------------------------------------------
Status of mpd:
Unit mpd.service could not be found.
 -) WARNING: mpd is not running as a service ... checking for processes
    WARNING: no processes matching mpd found
------------------------------------------------------------------
Status of pulseaudio:
 -) pulseaudio is running as a process:
    pi          1018     992  3 05:11 ?        00:00:00
/usr/bin/pulseaudio --daemonize=no --log-target=journal
------------------------------------------------------
IP address:  192.168.1.147
temperature: 58C / 136F
Memory usage:
...
CPU usage:   88%
```

If by chance there are two pulseaudio processes running it should be started by the user pi, but not by the user gdm. Add a line to both `/usr/lib/systemd/user/pulseaudio.socket` and `/usr/lib/systemd/user/pulseaudio.service` so that gdm can not start a second copy.

```
# sudo vi /usr/lib/systemd/user/pulseaudio.socket
...
ConditionUser=!root
ConditionUser=!gdm
...
# sudo vi /usr/lib/systemd/user/pulseaudio.service
...
ConditionUser=!root
ConditionUser=!gdm
...
```

Turn off Linux bluetooth

Getting bluetooth to work on Linux is quite complicated. The recommended amplifier comes with bluetooth that is very simple to use - you just turn the amp on and it works - no passwords. Therefore, it is recommended that you turn Linux bluetooth off with the following command.

```
$ sudo systemctl disable bluetooth
...
Removed /etc/systemd/system/bluetooth.target.wants/bluetooth.service.
```

After the next reboot, the bluetooth service should not be running.

Allow music to be read from a USB drive

A good source of music input is a USB drive with music in the form of MP3 or other music format files. Alternatively, you may have a network drive with such fiules.

If you have a USB drive with music on it, perform the following steps to have it mounted at GNU/Linux boot time. If you don't have one, you can skip this section, but you will need at least one source of music.

- Before you plug the USB drive into the RasPi USB jacks, check if there is anything mounted.

  ```
  $ mount | grep sda
  ```

  If you see no output, that means there is nothing mounted with the string 'sda' in it.

- Plug the USB drive with music on it into the RasPi. Then run the same **mount** command again. You should now see the USB drive device file.

  ```
  $ mount | grep sda
  /dev/sda1 on /media/pi/0011-5B59 type vfat (rw,nosuid,nodev,...)
  $ file /dev/sda*
  /dev/sda:  block special (8/0)
  /delv/sda1: block special (8/1)
  ```

  This shows that the USB drive was given the device name of `/dev/sda` and the first partition is `/dev/sda1`.

  Robert: you should see that the USB drive was auto-mounted over /media, you can skip the remaining steps in this section - mounting manually over /home/pi/usbdrive is extra work ...

- Unmount the drive so a better mount point can be chosen.

  ```
  $ sudo umount /media/pi/0011-5B59
  ```

- Allow all users to read the directory `/home/pi/`.

  ```
  $ cd /home
  $ ls -ld pi
  drwxr-x--- 20 pi pi 4096 Nov 19 12:16 pi
  $ sudo chmod 755 pi
  $ ls -ld pi
  drwxr-xr-x 20 pi pi 4096 Nov 19 12:16 pi
  ```

- Make a directory to mount USB drive over. In this example it is `/home/pi/usbdrive/`.

  ```
  $ cd
  $ mkdir usbdrive
  ```

  Now the USB drive can be mounted over that directory.

- Make a backup copy of the file `/etc/fstab`. This is the file that Linux reads to determine which file systems to mount at boot time.

  ```
  $ cd /etc
  $ sudo cp fstab fstab.orig
  ```

This is done because if you ever screw up `/etc/fstab`, your system may not be able to boot. Having a backup copy makes it easier to recover your system.

- Add one line to the bottom of `/etc/fstab` that wil mount the USB drive over the directory you just created.

```
$ sudo vi fstab
LABEL=writable  /        ext4   defaults        0 0
LABEL=system-boot       /boot/firmware  vfat    defaults 0 1
/dev/sda1               /home/pi/usbdrive  vfat nodev,nosuid,uid=1000,gid=1000 0 0
```

- Mount the file system by providing just one argument to the **`mount`** command. This effectively tests the change to the `/etc/fstab` file.

```
$ sudo mount /home/pi/usbdrive
```

- Show the contents of the USB drive.

```
$ ls -l /home/pi/usbdrive
drwxr-xr-x   2 root root 16384 Nov  2  2015 'System Volume Information'
drwxr-xr-x 222 root root 32768 May  9  2011  music
```

This shows a directory named `music` under which there are mp3 files.

- This step is optional, but you can reboot the system to test that the drive is mounted at boot time.

```
$ sudo reboot
```

Allow music to be read from Network Attached Storage

Another source of music is music files on Network Attached Storage (NAS). Add the following line to `/etc/fstab`:

```
# sudo mkdir /mnt/music
# sudo vi /etc/fstab
...
//192.168.1.64/MacisaacData /mnt/music cifs username=pi,password=pi,vers=1.0 0 0
# sudo mount /mnt/music
```

Install Emby

Robert: Skip this section - it will be delted once we get mpc/mpd to work ...

Emby is a music and video player. To install Emby, perform the following tasks.

- Point a browser to the Emby server download site.

```
https://emby.media/linux-server.html
```

- Copy the URL for Emby server package for the Arm64 architecture.
- Get the package with the **`wget`** command. For example:

```
$ wget
https://github.com/MediaBrowser/Emby.Releases/releases/download/4.7.5.0/emby-server-deb_4.7.5.0_arm64.deb
```

- Install the Emby server with the **`dpkg`** command.

```
$ sudo dpkg -i emby-server-deb_4.7.5.0_arm64.deb
Selecting previously unselected package emby-server.
(Reading database ... 163014 files and directories currently installed.)
Preparing to unpack emby-server-deb_4.7.5.0_arm64.deb ...
Unpacking emby-server (4.7.5.0) ...
Setting up emby-server (4.7.5.0) ...
Created symlink
/etc/systemd/system/multi-user.target.wants/emby-server.service ?
/lib/systemd/system/emby-server.service.
```

```
Processing triggers for libc-bin (2.35-0ubuntu3) ...
```

- Open a browser and set the URL to Emby using your IP address and port 8096. In this example it is:

  ```
  http://192.168.1.229:8096
  ```

- You should see a page with the title *Welcome to Emby.* Choose your language and click **Next**.
- At the *Create Your First User* page, set the Username and password to **pi** and click **Next**.
- At the *Setup Media Libraries* page, click the **+ New Library** button to add a new music folder. A *New Library* window should appear
  - In the *Content type:* text box, choose **Music**.
  - Click the **+** button next to *Folders*. Enter either **/home/pi/usbdrive** or **/mnt/music/music** and click **Ok**.
  - Back at the New Library page click **Ok**.
- At the *Setup Media Libraries* page, Click **Next**.
- At the *Preferred Metadata Language* page choose your language and country and click **Next**.
- At the *Configure Remote Access* page, Click **Next**.
- At the *Emby Terms of Service* page, accept the terms of service and click **Next**.
- At the *You're Done* page Click **Finish**.
- You should see a *Please Sing In* page. Choose the user **pi**.
- On the next page, enter the password for pi and click **Sign In**.
- You should see the Emby home page. Note the green dial in the music library you just created. This is the progress of Emby indexing your music.

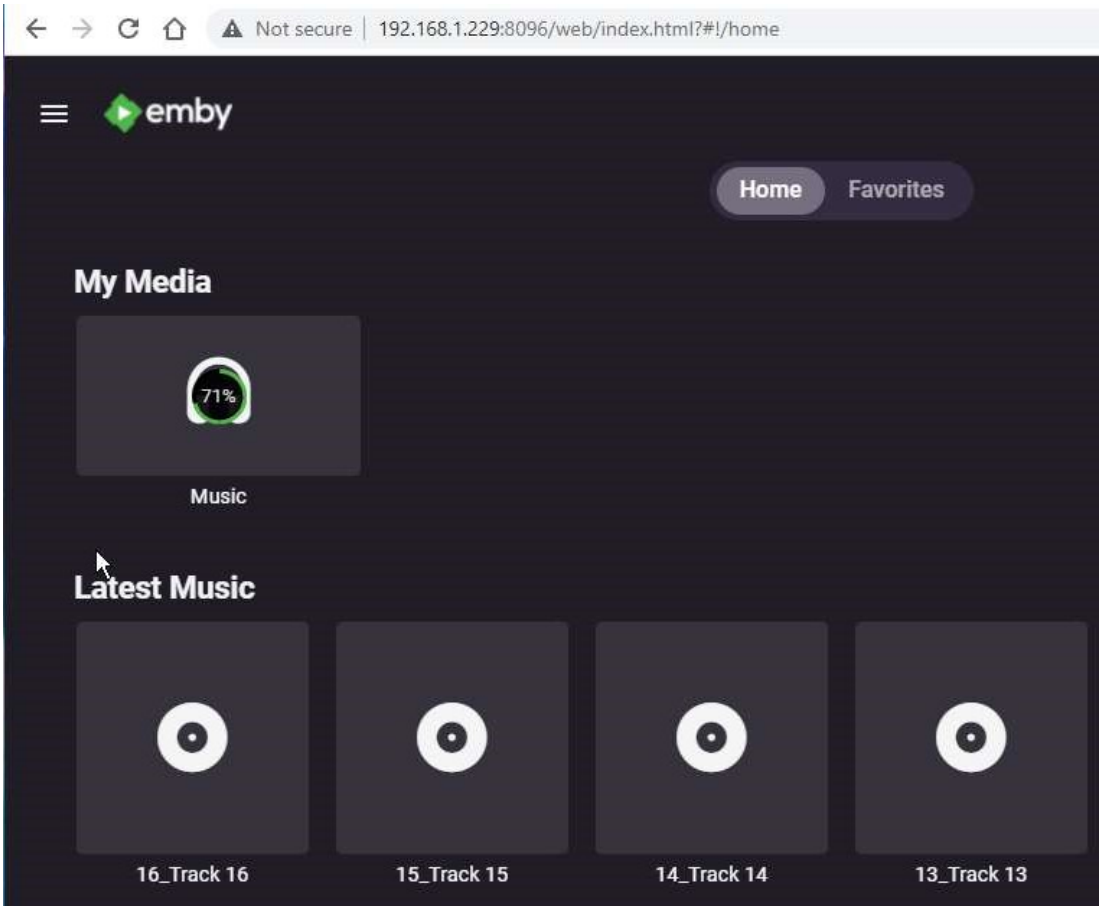

Figure 1-3 Emby home page

Emby should now be installed, and busy indexing your music.

Install a forked Mycroft mpc/mpd skill

Robert: Do this this section and skip everything Emby below.

Install mpc and mpd, then the forked skill so it can play music driven by voice.

- Install mpc and mpd.

```
$ sudo apt install mpc mpd
...
After this operation, 83.7 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
...
```

- Install python-mpd2 inside the Mycroft virtual environment.

```
$ source /home/pi/mycroft-core/venv-activate.sh
Entering Mycroft virtual environment. Run mycroft-venv-deactivate to exit
(.venv) pi@boombox1200:~$ pip install python-mpd2
...
Successfully installed python-mpd2-3.0.5
(.venv) pi@boombox1200:~$ mycroft-venv-deactivate

...
```

- Backup and edit the `/etc/mpd.conf` file. Most defaults are good.

- Set the music directory to `/media`
- Set the log file to `/var/log/mpd/mpd.log`
- Set the group to `audio`

```
$ cd /etc
$ sudo cp mpd.conf mpd.conf.orig
$ sudo vi mpd.conf
music_directory         "/media"
playlist_directory      "/var/lib/mpd/playlists"
db_file                 "/var/lib/mpd/tag_cache"
state_file              "/var/lib/mpd/state"
sticker_file            "/var/lib/mpd/sticker.sql"
log_file                "/var/log/mpd/mpd.log"
user                    "mpd"
group                   "audio"
bind_to_address         "localhost"
input {
        plugin          "curl"
}
decoder {
        plugin          "hybrid_dsd"
        enabled         "no"
}
decoder {
        plugin          "wildmidi"
        enabled         "no"
}
filesystem_charset      "UTF-8"
```

- Be sure there is world read access on `/media/` down to the directory where the USB drive gets automounted.

```
$ ls -ld /media /media/pi/ /media/pi/2F11-AF89/
drwxr-xr-x  3 root root  4096 Dec 21 18:08 /media
drwxr-xr-x+ 3 root root  4096 Dec 22 06:27 /media/pi/
drwxr-xr-x  5 pi   pi   32768 Dec 31  1969 /media/pi/2F11-AF89/
```

Robert - was this needed?

```
$ sudo chmod o+rx 2F11-AF89
```

- Enable **`mpd`** to start at boot time.

```
$ sudo systemctl enable mpd
Synchronizing state of mpd.service with SysV service script with
/lib/systemd/systemd-sysv-install.
...
```

- Reboot to test that **`mpd`** starts.

```
$ sudo reboot
```

- Restart an SSH session and run the **`lsenv`** command. You should see mpd running, but not mycroft.

```
$ lsenv
Status of mycroft:
 -) WARNING: mycroft is not running as a service ... checking for
processes ...
    WARNING: no processes matching mycroft found
------------------------------------------------------------------
Status of emby-server:
Unit emby-server.service could not be found.
 -) WARNING: emby-server is not running as a service ... checking for
processes ...
```

```
    WARNING: no processes matching emby-server found
-------------------------------------------------------------
Status of mpd:
 -) mpd is running as a service:
    Active: active (running) since Thu 2022-12-22 06:27:10; 2min 47s ago
-------------------------------------------------------------
Status of pulseaudio:
 -) pulseaudio is running as a process:
    pi          1048     999  0 06:27 ?        00:00:00
/usr/bin/pulseaudio --daemonize=no --log-target=journal
-------------------------------------------------------
IP address:  192.168.1.147
temperature: 55C / 131F
Memory usage:
...
```

- Tell mpd to index the music. Then **`mpc listall`** should show all your song files.

```
$ mpc update
$ mpc listall
...
```

- Open a wide SSH session and start mycroft in dev/debug mode. You should see a console come up with log files, mycroft interactions and a command input section.

Start Mycroft

```
$ mycroft-start debug
```

- Type in the input section what time is it. Mycroft should speak the time.

```
============================== Mic Level ==
 what time is it
 >> It's six thirty five
```

- Clone the skill from github in pi's home directory.

```
$ cd
$ git clone https://github.com/mike99mac/mpc-skill-mike99mac
...
```

- Recursively copy the new directory to the Mycroft skills directory. If you have the mycroft console open, you should see it load that skill.

```
$ cp -a mpc-skill-mike99mac /opt/mycroft/skills
```

Install the Mycroft Emby skill

NOTE: this is Mike's forked Emby skill which will hopefully die a natural death:

```
$ git clone https://github.com/mike99mac/emby-skill-mike99mac
```

Now install the related Mycroft skill so Emby can be driven by voice.

- Install Emby with the **`mycroft-msm`** command and verify it was installed successfully.

```
$ mycroft-msm install emby
INFO - building SkillEntry objects for all skills
INFO - Best match (1.0): emby by rickyphewitt
...
INFO - Successfully installed emby
```

Update the Emby skill

The Emby skill has important values such as the URL, user ID and password of the Emby server. To configure it, perform the following steps:

- Go to the Mycroft skills page:

  `https://account.mycroft.ai/skills`

- A page should open with the title *Configure Your Skills*. You should see the Emby skill. Click the down arrow to the right.
- Set the URL, user and password appropriately and click **SAVE** as shown in the following figure.

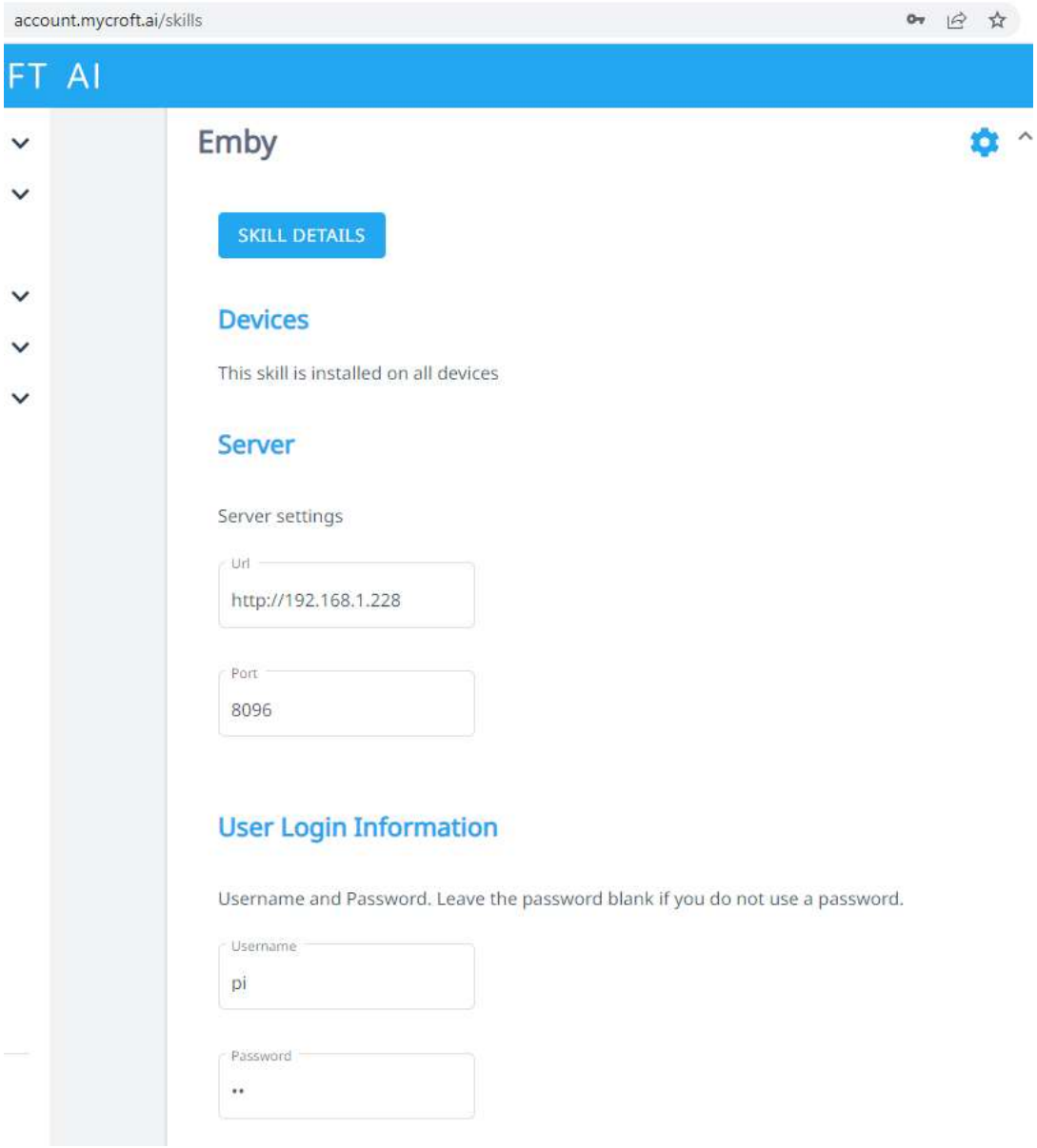

Figure 1-4 Emby skill settings

- Open a browser and set the URL to Emby using your IP address and port 8096. In this example it is:

  `http://192.168.1.228:8096`

Install Mycroft radio

Mycroft radio allows radion stations to be streamed from the Internet.

Robert: skip this too - this skill intercepts music requests and causes conflation!

To install Mycroft radio, perform the following tasks.

- Clone it from github with the following command.

```
$ git clone https://github.com/MycroftAI/skill-mycroft-radio
Cloning into 'skill-mycroft-radio'...
remote: Enumerating objects: 232, done.
remote: Counting objects: 100% (232/232), done.
remote: Compressing objects: 100% (156/156), done.
remote: Total 232 (delta 34), reused 217 (delta 23), pack-reused 0
Receiving objects: 100% (232/232), 399.35 KiB | 3.99 MiB/s, done.
Resolving deltas: 100% (34/34), done.
```

- When the system comes back up, start an SSH session as pi and check that the USB drive is mounted.

```
$ mount | grep usbdrive
/dev/sda1 on /home/pi/usbdrive type vfat (rw,...)
```

NOTE: If you can hear music, but the volume is low, you can increase the gain using **`alsamixer`**. To do so, perform the following steps

- Start **`alsamixer`**.

  ```
  $ alsamixer
  ```

  You should see basic graphics as shown in Figure 1-5 on page 31.

- Use the right arrow key to move to *Digital* gain.
- Use the up arrow to increase the gain. As you press the key, the display and the volume should both increase.
- Press the **Esc** key to exit alsamixer. Your changes will be saved.

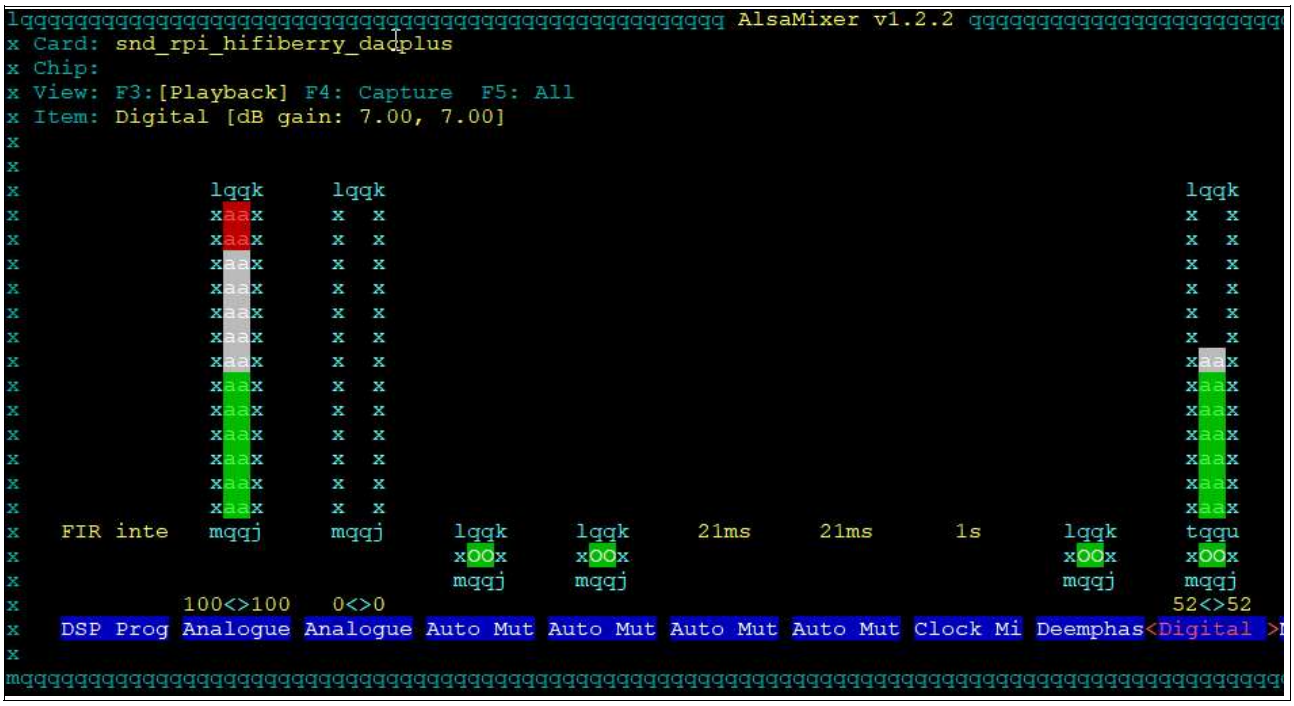

Figure 1-5 Alsamixer Digital gain adjustments

Update the pi user

Now that pulseaudio and Jellyfin are installed, those groups can be added to the `pi` user.

- Add `pulse` and `emby` as alternate groups to the user `pi` with the following command.

```
$ sudo usermod -a -G pulse pi
$ sudo usermod -a -G emby pi
```

1.8.1 Use the ??? music streaming service

Note: If/when Mycroft partners with a music provider, document here ...

1.8.2 Use Internet radio

To access Internet radio, perform the following steps.

- Change to the skills directory and install the `mycroft-skill-tunein` skill.

```
$ cd /opt/mycroft/skills
$ git clone https://github.com/MycroftAI/skill-mycroft-radio
Cloning into 'skill-mycroft-radio'...
...
```

-

1.8.3 Installing Mimic3

To install Mimic3, perform the following steps.

- Install the required `python3.10.venv` and `python-pip` packages.

```
$ sudo apt-get install -y python3.10-venv python3-pip
...
```

- Change to the home directory and install Mimic3 from github.

```
$ cd
$ git clone https://github.com/MycroftAI/mimic3
Cloning into 'mimic3'...
...
Resolving deltas: 100% (1047/1047), done.
```

1.9 Development environment

There is an excellent browser-based development system. The skill was created by @andlo.

To install it, perform the following.

```
$ cd $SKI
$ mycroft-msm install https://github.com/andlo/openvscode-server-skill.git


Adding other skills:
git clone https://github.com/andlo/count-skill
git clone https://github.com/pcwii/skill-pick-number
git clone https://github.com/Dragoncraft89/dice-skill
git clone https://github.com/padresb/bark-skill
git clone https://github.com/MycroftAI/skill-stop
git clone https://github.com/MycroftAI/fallback-duckduckgo
git clone https://github.com/MycroftAI/skill-query

--------------------------------------------------
Testing internet radio:
git clone https://github.com/normandmickey/skill-internet-radio

==> It appears this skill does not work and is no longer being maintained
--------------------------------------------------

... Trying "Tune In" for Internet radio stations

$ cd ~/mycroft-core
$ source .venv/bin/activate
(.venv) $ msm install https://github.com/johnbartkiw/mycroft-skill-tunein

TODO: easier - try
mycroft-msm install https://github.com/johnbartkiw/mycroft-skill-tunein


https://sourceforge.net/projects/win32diskimager/

------------------------------------------------------------
Ideas on selling graduated kits

SmartBoombox 0.2 - just the wood to assemble an enclosure - you glue up
and supply all hardware
SmartBoombox 0.4 - an assembled enclosure - you supply all hardware, but
no gluing
SmartBoombox 0.6 - an assembled enclosure and all audio components - does
not have to be smart
SmartBoombox 0.8 - an assembled enclosure and all parts to construct a
smart boombox
SmartBoombox 1.0 - a fully-assembled smart boombox

--------------------------------------------------
TODO: look into "Share a Cart" Chrome extension to share an Amazon
shopping cart

   ---------------------------------------------------
   Plywood replacement - avoid extra gluing
   Baltic Birch 12mm x 24" x 30"
```

```
https://www.woodcraft.com/products/baltic-birch-plywood-12-mm-1-2-x-24
-x-30
```

All software and hardware should now be installed and ready to test

1.9.1 How to clone from a Mycroft branch

It was suggested that I 'check out the mark-ii/qa branch'. Here's what I did as the user `pi`:

```
$ cd
$ sudo service mycroft stop
$ rm -fr mycroft-core/
$ git clone -b mark-ii/qa https://github.com/MycroftAI/mycroft-core
Cloning into 'mycroft-core'...
...
Resolving deltas: 100% (544871/544871), done.
```

1.9.2 Trying the OVOS common play code

```
Trying
$ pip install ovos_plugin_common_play
```

1.10 Test and enjoy your boombox

1.10.1 Testing the basic boombox

If you built the basic boombox, testing is much simpler. For music input you have a choice of using the aux-in jack or a bluetooth connection.

1.10.2 Testing the smart boombox

Start with **`testrecord`** and **`testplay`**.

Questions?

If you have any questions, feel free to email `mike99mac@gmail.com`

1.11 Music playing Mycroft skills

How does this stuff work??????????????????

Table 1-2 Important music playing skill source files

Source file	LoC	Class(es) defined, description
/opt/mycroft/skills/mycroft-playback-control.mycroftai/__init__.py	307	PlaybackControlSkill common language for audio playback skills
~/mycroft-core/mycroft/audio/audioservice.py	551	AudioService
~/mycroft-core/mycroft/skills/audioservice.py	168	AudioService ??? why two?
~/mycroft-core/mycroft/skills/common_play_skill.py	310	CommonPlaySkill, CPSMatchLevel, CPSTrackStatus
/opt/mycroft/skills/emby.rickyphewitt/__init__.py	165	Emby(CommonPlaySkill) Mycroft skill hooks
/opt/mycroft/skills/emby.rickyphewitt/emby_client.py	237	PublicEmbyClient, EmbyClient(PublicEmbyClient) An intentionally lean synchronous Emby client
/opt/mycroft/skills/emby.rickyphewitt/emby_croft.py	320	EmbyCroft(object) Logic layer between Emby client and Mycroft

Functions in emby_croft.py

```
def from_string(enum_string):
def find_file(res_name, _=None, lang=None):
def __init__(self, host, username, password, client_id='12345',
diagnostic=False):
def determine_intent(intent: dict):
def handle_intent(self, intent: str, intent_type: IntentType):
def find_songs(self, media_name, media_type=None)->[]:
def search_artist(self, artist):
def search_album(self, artist):
def search_song(self, song):
def search(self, query, include_media_types=[]):
def get_instant_mix_songs(self, item_id):
def instant_mix_for_media(self, media_name):
def get_albums_by_artist(self, artist_id):
def get_songs_by_album(self, album_id):
def get_songs_by_artist(self, artist_id):
def get_all_artists(self):
def get_server_info_public(self):
def get_server_info(self):
def convert_response_to_playable_songs(self, item_query_response):
def convert_to_playable_songs(self, songs):
def parse_search_hints_from_response(response):
def parse_response(response):
def parse_common_phrase(self, phrase: str):
def set_version(self):
def normalize_host(host: str):
def diag_public_server_info(self):
```

Functions in emby_client.py

```
def __init__(self, host, device="noDevice", client="NoClient",
client_id="1234", version="0.1"):
def get_server_info_public(self):
def __init__(self, host, username, password, device="noDevice",
client="NoClient", client_id="1234", version="0.1"):
def _auth_by_user(self, username, password):
def get_headers(self):
def search(self, query, media_types=[]):
def instant_mix(self, item_id):
def get_song_file(self, song_id):
def get_albums_by_artist(self, artist_id):
def get_songs_by_album(self, album_id):
def get_songs_by_artist(self, artist_id, limit=None):
def get_all_artists(self):
def get_server_info(self):
def _post(self, url, payload):
def _get(self, url):
def __init__(self, user_id, token):
def from_response(cls, response):
def __init__(self, id, name, type):
def from_item(cls, item):
def from_list(items):
def from_string(enum_string):
```

Functions in emby_client.py

```
def __init__(self):
#   def find_file(res_name, _=None, lang=None):
def initialize(self):
def handle_emby(self, message):
def speak_playing(self, media):
def handle_diagnostic(self, message):
def stop(self):
def CPS_start(self, phrase, data):
def CPS_match_query_phrase(self, phrase):
def connect_to_emby(self, dagnostic=False):
def create_skill():
```