Raspberry Pi/Raspian/Python Notes

Last edited:

This document is a valuable tool for me to learn and reference specific findings I have made in the Raspberry Pi domain. As this domain is undergoing constant change and because I have a lot to learn, this document is a work in process. I also need to have the document available on my primary PC, a laptop, and even a Raspberry Pi computer, I plan to keep a master under version control on GitHub in a 'Notes' project. Document updates will be updated on the current computer in use and committed to the GitHub master repository.

These notes originated in 2015 for use on a Raspberry Pi 2 Model B v1.1 using the latest NOOBS package at the time. The current notes are written for NOOBS version 1.9.2, Released 2016-05-27 and reflect changes that have evolved over a year. If notes don't seem to fit what is viewed on the RPi screen, suspect a revision in the ever evolving software in use.

Topics:

1. Installing operating system images. Start here:

<https://www.raspberrypi.org/documentation/installation/installing-images/README.md>

and here:

<https://www.raspberrypi.org/documentation/installation/installing-images/windows.md>

and furthermore: NOOBS Setup at

<https://www.raspberrypi.org/help/noobs-setup/>

Note Win32DiskImager is already installed. Look for application named "Image Writer".

**Configuration list (for new systems on SD memory.)**

* 1. Using NOOBS on newly formatted SD memory card
     1. Specify WiFi router pass code.
     2. Install Raspbian.
  2. In GUI Preferences, use RPi Configuration tool to enable camera, Set Localization items.
  3. Copy key development files from old SD memory to new SD card.
  4. Perform sudo upgrade update installs for camera utilities, GPIO. (Examples: beware some instructions may be obsolete or buggy as experienced in raspberrypi.org)

Camera: <https://www.raspberrypi.org/documentation/configuration/camera.md>

1. Command line control of RPi.
   1. Using a remote laptop as terminal.
   2. Log-in using username/password. Default is "pi"/"raspberry"
   3. Starting the Raspian GUI.
      1. Enter startx on the command line followed by Enter. Automatic entry into the GUI may occur if Preferences.RaspberryPiConfiguration is configured to do this.
   4. Shutting down safely.
      1. From the GUI.
         1. Click on File.Exit.Shutdown.
      2. From the CLI. *sudo shutdown -h now*
         1. For hardware/Python method visit:
   5. Mounting a USB thumb drive to transfer files.
      1. CLI approach
      * Find out what the USB drive is called: sudo fdisk -l (For this example returns */dev/sda1*)
      * Create a mount point: sudo mkdir /media/usb
      * Mount: sudo mount */dev/sda1* /media/usb
      * Do work (copy files two or from USB drive)
      * Unmount: sudo umount /media/usb
      * Credit <http://askubuntu.com/questions/37767/how-to-access-a-usb-flash-drive-from-the-terminal-how-can-i-mount-a-flash-driv>.
      * Error saying disk is already mounted? Use df –h to get list of mounted drives and paths. Study the paths to find where to access the thumb drive.
      1. GUI approach
      * Plug USB drive in open slot.
      * Use File Explorer to find USB contents in /media/pi/root
      * Do stuff (copy from-to)
      * Eject from icon in upper right of GUI. (Click on triangle-over-bar icon to get a list of media along with their folders in /media/pi (boot, root, SETTINGS)

1. Configuration of Raspbian:
   1. From the CLI - sudo raspi-config
   2. From the GUI - See Preferences
2. Display a list of running processes: *ps ax*
3. Display the current SSH version: *ssh -V*
4. Other key command line commands: See [Debian Refcard.pdf](DebianRaspian%20OS/Debian%20Refcard.pdf)
5. Capturing history of all command lines to file: history 1>historyfile.txt See [redirection](http://www.tldp.org/LDP/abs/html/io-redirection.html) for more details.
6. Taking a screen shot in CLI: *scrot*.
   1. scrot with x second delay: scrot *-d x*
   2. scrot with countdown delay: scrot *-cd x*
   3. scrot for currently active window: scrot *-u -cd x*
   4. *scrot to output file (date, time, resolution, scrot) and move to receiving file:*
   5. *scrot '%Y-%m-%d*\_%H%M *\_$wx$h\_scrot.png' -e 'mv $f ~/images/shots/'*
7. Locating packages for my unique needs:
   1. Look in Debian Packages for jessie: <https://packages.debian.org/stable/>
   2. To search available packages for a keyword, use 'search' plus the keyword of interest:

*apt-cache search <keyword>* or *apt-cache pkgnames <specific package name>*

1. Finding and Installing packages: *sudo apt-get <package name>*. Optionally, add '*-Y*' to automatically say Yes to questions that may arise.
2. List installed packages:
   1. *dpkg --get\_selections > dpkg-selections.txt* (OR *| less*),
   2. OR *dpkg --list* (mentioned elsewhere in this document)
   3. Alternate: [synaptic](https://thepihut.com/blogs/raspberry-pi-tutorials/16998732-finding-out-what-is-installed-on-your-pi-and-installing-more) (install with *sudo apt-get install synaptic*)
3. Removing packages:
   1. *sudo apt-get purge <package name>* to remove main package
   2. *sudo apt-get autoremove* to uninstall dependencies that were added with the purged package.
4. Creating and running Shell Scripts. See [Advanced Bash-Scripting Guide](http://www.tldp.org/LDP/abs/html/why-shell.html). Also, see snappic.sh example below.
   1. First line is required, this is called the 'shebang':

#! /bin/bash

* 1. Make script executable: *chmod +x <script name>.sh*

1. Running from the command line should be done by typing "./snappic.sh" if already in the folder holding the script. Leaving the ./ prefix off doesn't run the script! When script development is complete, copy it with a "sudo cp" to /usr/local/bin to eliminate "./" requirement. See [Invoking the Script](http://www.tldp.org/LDP/abs/html/invoking.html).
2. Scripts can invoke other scripts as long as the absolute path is defined (or it is in /usr/local/bin)
3. The script can run on a schedule using crontab instructions below. No further interaction needed from power-up.
4. Saving data in the cloud: See example for [Dropbox Sync](https://learn.adafruit.com/cloud-cam-connected-raspberry-pi-security-camera/dropbox-sync) and [Adafruit IO](https://learn.adafruit.com/cloud-cam-connected-raspberry-pi-security-camera/adafruit-io) in adafruit website. When installing Dropbox Uploader, it will be put your Documents directory in its own folder as: /home/pi/Documents/Dropbox-Uploader. For additional guidelines, see other Dropbox setup instructions in the Dropbox Sync link above. Following these instructions, you can create a unique Dropbox folder for each RPi instance or give yourself access to all your DB folders. For organization purposes, the following installations are recorded:

Media App name Dropbox Folder

* 1. MicroSD 32GB for RPi 3 RPi3Uploader (?) Dropbox/Apps/RPiPictures
  2. MicroSD 32GB for RPi 3 RPi3Uploader8977 Full Dropbox access, default to Dropbox/Apps/RPiPictures
  3. /MicroSD 8GB for RPi ZERO v1.3 Dropbox/Apps/RPiJunk

Type *dropbox\_uploader* on CLI for usage instructions. Sample upload from CLI when Full Dropbox access is used (this is a project folder backup): *dropbox\_uploader upload ~/Documents/PythonProjects/ /Apps/RPiPictures/* (trailing '/' is important)

1. Installing the camera. This assumes RPi is booted into GUI vs CLI (Command Line Interface).

Background: <https://www.raspberrypi.org/documentation/usage/camera/> and

<https://www.raspberrypi.org/documentation/raspbian/applications/camera.md>

* 1. Start CLI window, a.k.a. "The Terminal".
  2. sudo apt-get update
  3. sudo apt-get upgrade
  4. exit to GUI
  5. Start Preferences.RaspberryPiConfiguration dialog from GUI and enable camera.
  6. Quit Config dialog. Reboot if necessary.

The camera can be used from Shell scripts or a Python program. For shell scripts, see: <https://www.raspberrypi.org/documentation/usage/camera/raspicam/README.md>

* 1. From Python 3, see examples in [RaspberryPi.org Getting Started with Pi Camera](https://www.raspberrypi.org/learning/getting-started-with-picamera/worksheet/). Coding is relatively easy! Also: <https://www.raspberrypi.org/documentation/usage/camera/python/README.md>
  2. From Shell script (created in CLI window): raspistill, raspivid, Time-Lapse, raspiyuv.
  3. For Time-Lapse collection see: <https://www.raspberrypi.org/documentation/usage/camera/raspicam/timelapse.md> for description of cron and crontab usage. Open for editing: crontab -e and the LeafPad editor for snappic.sh files.

Take a time-stamped picture with the shell script *snappic.sh* listed here*:*

#! /bin/bash

DATE=$(date +"%Y-%m-%d\_%H%M")

raspistill -o /home/pi/pictures/$DATE.jpg

Use the LeafPad editor to open snappic.sh for editing. Folder Documents/MyProjects

Edit crontab to schedule snappic.h. Do crontab -e to open for edit and append this:

# m h dom mon dow command

# Camera Stills

\*/2 6-20 \* \* \* /home/pi/Documents/MyProjects/snappic.sh 2>&1

Which translates to:

"every 2 minutes, between 6am & 8pm (Standard time?), run snappic.sh, redirect stderr to stdout"

**To disable the time lapse script, comment out the crontab line containing snappic.sh with a "#".**

Set execute permission for scripts in MyProjects with the command:

chmod 777 -R /home/pi/Documents/MyProjects

or set Properties.Permissions for the folder in the GUI.

Can list crontab contents with the command: crontab -l

Get help with crontab with ^G after opening cronab -e.

1. Creating a time-lapse movie using RPi.

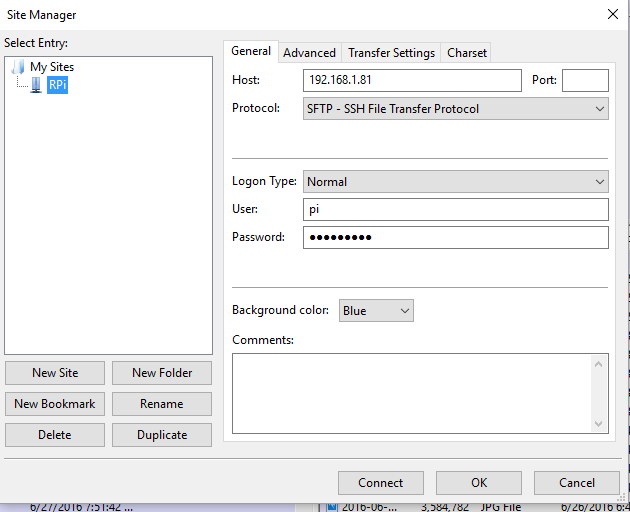
See Evernote>"TimeLapse Videos to iPad via a PC" for RPi notes.

1. Copy stills from microSD to C:\My Documents\Scratch.  Use FileZilla client on Windows PC or USB drive. (USB drive command is second best. Need to mount the drive on RPi with "sudo mount -t vfat /dev/sda1 /mnt" )

<http://trevorappleton.blogspot.com/2014/03/remotely-copy-files-to-and-from-your.html>

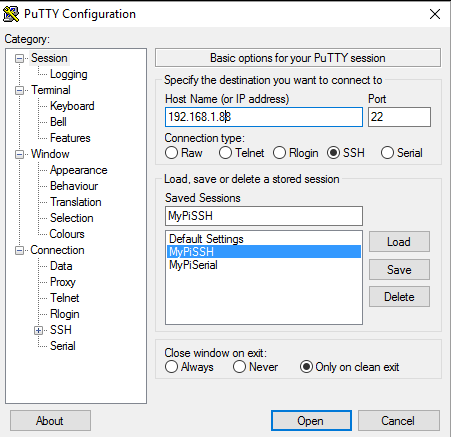
File.SiteManager screen looks like this: (note "Port" is blank) Host IP could be Ethernet or WiFi. If displayed, click through the warning of "unknown host key".

Host IP (for RPi) can be displayed in GUI window by hovering over the wireless icon in upper right corner. Note: the IP address may change if power has been recycled on the RPi!



1. Use Movie Maker (Windows Movie Maker, Version 2012 used here) to make .mp4 movie segments from batches of pictures (349 images in a batch has worked when 16GB RAM is available on the PC).  0.04 secs/frame.   Select and clear all stills after each batch.   Note where movie segments are stored: ThisPC/Videos. Reference: <http://www.instructables.com/id/How-To-Make-a-Film-with-Pictures-using-Windows-Mov/?ALLSTEPS>
2. If necessary, use Movie Maker to combine .mp4 movie segments.   Add segments to workspace, set same frame rate, & make movie.
3. Optionally, move output from This PC/Videos folder to Photos/ iPad Pics folder
4. Use iTunes to Sync Photos (including videos).
5. Watch video on iPad device.
6. Create an animated GIF using the ImageMagick package. This is described in Instructable *Raspberry Pi Class: Lesson 5 Navigate the RPI's Software: Part 2*
   1. Add these two lines to a shell script text file, <script name>.sh: 
   2. convert is a command from ImageMagick. Takes all saved jpegs withthe prefix of image and converts them into an animated GIF with a delay of 10/100 of a second and loops forever.
   3. Make the shell script executable with the *chmod +x <script name>.sh*

* Wireless terminal connection to RPi (SSH)
  1. This assumes the target computer has ethernet or WiFi access.
  2. SSH (Secure Shell) is a function provided in the application PuTTY. This free .exe program is found on Pavilion at "C:\Users\Alan\Documents\Raspberry Pi\PuTTY\putty.exe".
  3. Obtain the target computer IP address. On my network this can be done with the FING App.
  4. Start PuTTY and enter the IP address where indicated. Select SSH access to the target. Click 'Open'. A typical configuration for PuTTY is illustrated here:



* 1. To end a SSH session in PuTTY: Just exit PuTTY using the X box inupper right. Other methods on web did not work for me: *Ctrl-D* and *~.* ***~.*** means 'disconnect' per man 1 ssh
* Configuring a RPi (or ZERO) to run without keyboard or monitor ("headless"). See this excellent procedure [Raspberry Pi Zero Headless Setup](https://davidmaitland.me/2015/12/raspberry-pi-zero-headless-setup/), 1 Dec2015 on DavidMaitland.me. Used to configure Zero for wireless operation over SSH with Laptop. Procedure written for use on a Linux (RPi) system. To save space here, I converted the webpage to PDF and stored in Documents/Raspberry Pi/RPiHints/Raspberry Pi Zero Headless Setup.pdf.
* Configuring a RPi to start a python script on startup. Run the program on boot with a special crontab command.
* How to check if a program is running. 'ps' lists all running programs started from the command line. 'ps axq' lists all programs including all those owned by the system. **'ps axg | grep <prog name>'** filters out all in list that don't have the program name included.
* Scripts to automate cropping and saving image files.
* Scripts to do OCR on meter readout.
* Loading RPi.GPIO for RaspiRobot

1. What is loaded with this distribution of Debian > Raspian > Wheezy?
   1. dpkg --list
   2. dpkg --search <function>. <function> example: "python-rpi.gpio"
   3. aptitude is a friendlier package manager. Run 'aptitude' from command line to get to GUI. Review Help to get started.
2. **Transferring files between RPi and Windows**. Ans: [FileZilla](http://trevorappleton.blogspot.com/2014/03/remotely-copy-files-to-and-from-your.html). Use ifconfig to get RPi iP address. Fill Quickconnect fields for Host (RPi IP address), Username (pi), Password (raspberry) and Port (no entry) before clicking on Quickconnect button. Browse in the 'Remote site:' window to the folder holding files of interest, select one or more files, right-click and select
3. **Search disk for a file name**.Use the ‘tree’ command. For more info do ‘info tree’ or google up linux tree. Some useful commands.
   1. tree ~ : show all directories in my user account.
   2. tree ~ -d
   3. tree / -P ‘snappic.sh’ –prune: search entire disk for snappic.sh, don’t show folders without the file. NOTE prune has double hyphens which Word insists be shown as a long hyphen. “ is equivalent to ‘ in the options string.
   4. Similarily: tree –P ‘cron\*’ / --prune displays all folders with files containing ‘cron’ and variations. No asterisk will look for exact match to cron.
   5. Tree ~ -o myfolders.txt : sends output to myfolders.txt
   6. Tree ~ –P ‘\*’ –o myfiles.txt -s –D sends listing of all files in user account to a file. Include file size and date of last modification. Do info tree!
4. **Search disk for a file name**: Alternative to tree: find ~ -name <filename> (Do info find). Use “–iname” to ignore case.
5. Installing a package written for Python 2.7 in Python 3.4. See example cited below for adafruit/Adafruit Python ADS1x15 package.
6. Testing Internet connectivity with a Python script. Use Python program testinet.py to test connectivity to Google network. Results logged to screen and the file: inettest.txt. Data logged on one line per test: date/time/message/number of tests made/number of tests failed.

# Packages I installed (name, source, special instructions)

1. Weather forecast data
2. ADC module for ADS1115 16-bit 4-channel ADC with Prog. Gain Amps. From Adafruit Learn topic [Raspberry Pi Analog to Digital Converters by Tony DiCola](https://learn.adafruit.com/raspberry-pi-analog-to-digital-converters/mcp3008). Installed from source on GitHub at [adafruit/Adafruit\_Python\_ADS1x15](https://github.com/adafruit/Adafruit_Python_ADS1x15). At time of install the instructions specified python which install the package for Python 2.7, modified instructions below to install for Python 3.4.

sudo apt-get update

sudo apt-get install build-essential python-dev python-smbus git

cd ~

git clone https://github.com/adafruit/Adafruit\_Python\_ADS1x15.git

cd Adafruit\_Python\_ADS1x15

sudo python setup.py install <= Replace 'python' with 'python3' here.

Per Lutz *Learning Python* p 681-682, verify with Python 3.4 interactive commands: >>> import sys and >>>sys.path to list the current path.

While the simple change in instructions above seems to work, this is a learning opportunity creating setup.py scripts and for how to define PYTHONPATH per Lutz *Learning Python* Chapter 22 "Module Search Path" & Appendix A section on "Configuring Python". Also web resources (for starters search "Editing PYTHONPATH" in Raspberrypi.org, "How to permanently set environmental variables" and "Permanent Environment Variable for all users" in stackexchange)

1. GUI on Linux - 'mc' (installed on Zero2)
2. Matplotlib 1.5.3 from [matplotlib.org](http://matplotlib.org/).
3. PIXEL fromRaspberryPi.org - hold off on other PIs! There are questions asked without much guidance for what is needed.

# Physical interface APIs

**GPIO ZERO**

Introduction: <https://www.raspberrypi.org/blog/gpio-zero-a-friendly-python-api-for-physical-computing/>

# Python Notes

References:

* Ref A: Python 3.4 Tutorial <https://docs.python.org/3.4/tutorial/index.html>
* Ref B: Python Tutorial: <http://www.tutorialspoint.com/python/index.htm>
* Ref C: Python.org <https://www.python.org/doc/>
* Ref D: Instructables [Raspberry Pi Class](http://www.instructables.com/class/Raspberry-Pi-Class/?utm_source=classesNL&utm_medium=email) with push\_reset

1. How to see what names are defined by a [module](https://docs.python.org/3.4/tutorial/modules.html#modules)? In the Python interpreter, type 'dir(modulename) to get a list of names in the module (properties and methods).
2. What names are defined currently in the interpreter? Type 'dir()'.
3. How to get documentation for names? dir(modulename) yields properties and methods. help(modulename) yields description of names, help(modulename.name) yields brief help for name.
4. To list names of built-in functions and variables, import the module builtins and type dir(builtins)
5. Python interactive command history is stored in user directory in file .python\_history. (Verify this)
6. GUI for my applications: use tkinter package. Introduced in Programming in Raspberry Pi (Monk). Use dir(Tk) and help(Tk) or help(Tk.<name>) or help(Wm) to find out details, as Monk's book is short on explanation. Note: Class Wm in tkinter, provides functions for communication with the window manager.
7. Defining long strings: use triple quotes ''' or """. Long strings can span multiple lines and can include special characters (NEWLINEs, TAB, etc.) Ref B:
8. Learning module and class methods and data? ans: Google search *'How how I look inside a Python object?*' and going to StackOverflow.com: Working in Python, first import the module, then

in Python command line (or shell), type: dir(functionname)

OR help(module or module.class)

OR: Python has a strong set of introspection features.

Take a look at the following [built-in functions](http://docs.python.org/library/functions.html):

* type()
* dir()
* id()
* getattr()
* hasattr()
* globals()
* locals()
* callable()

type() and dir() are particularly useful for inspecting the type of an object and its set of attributes, respectively.

**OR**: or object.\_\_dict\_\_

**OR**: to see actual source code:

>>> import string

>>> string.\_\_file\_\_

'/usr/lib/python2.5/string.pyc'

The \*.pyc file is compiled, so remove the trailing 'c' and open up the uncompiled \*.py file in your favorite editor or file viewer:

/usr/lib/python2.5/string.py

I've found this incredibly useful for discovering things like which exceptions are raised from a given API. This kind of detail is rarely well-documented in the Python world.

1. Networking in Python.

Learning Python with Raspberry PI, Networked Python. Program code can be downloaded from <www.wiley.com/go/pythonraspi>

What is my IP address? Ans. LXTerminal command ifconfig. Alternate: use iPad FING app!

1. Sending images to the cloud. See above "Saving data to the cloud" for Dropbox and Adafruit IO examples.

# Git Notes

See my separate document on this topic entitled Git-WorkFlowForNewRepository.docx