

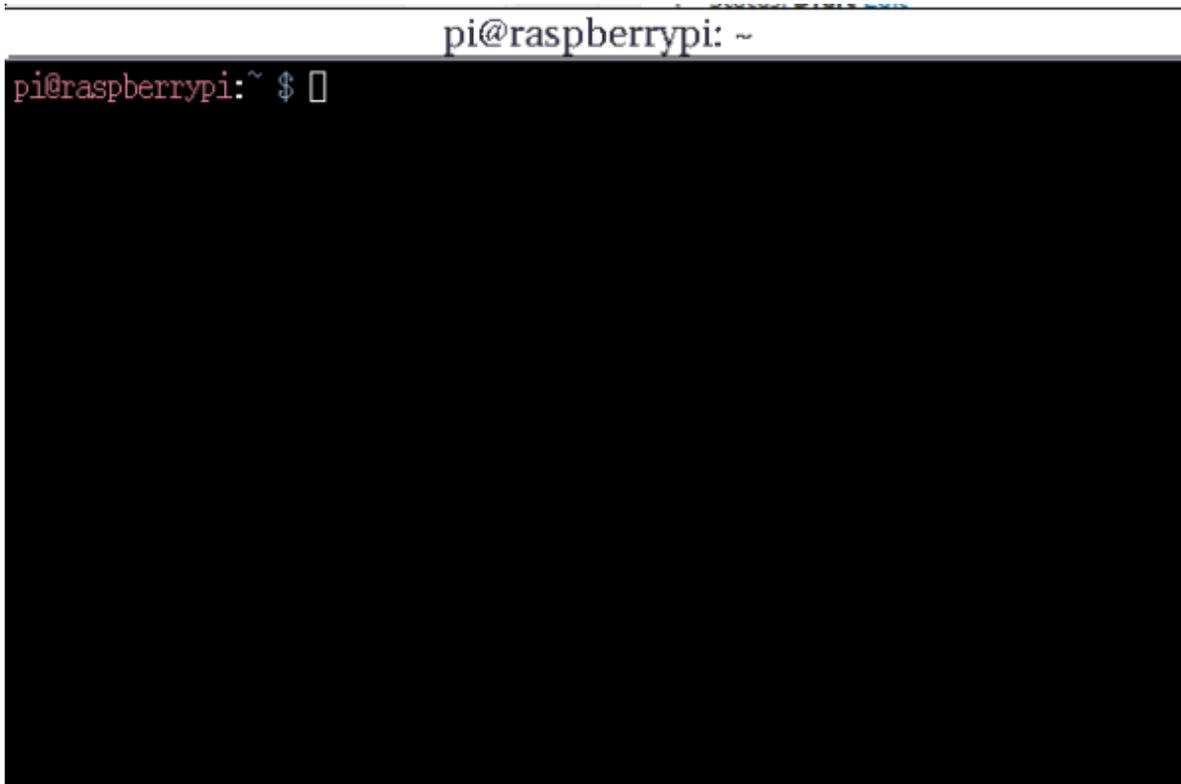
# RPi: Terminal + Python Intro

## Navigating with the terminal

Like on windows or any other computer, files in Linux are arranged in a *hierarchical directory structure*; meaning starting at root there is a sort of tree of folders containing files and other folders.

The terminal has a concept of 'working directory' ; using the metaphor of navigating an office building this is simply what room the terminal is in. If you use a command to list files it will look around the current directory and return you a list of files currently visible.

After opening a terminal with the icon in the upper left you will be confronted with something like this (but most likely with a different color-scheme and font)



```
pi@raspberrypi: ~
pi@raspberrypi:~ $ █
```

upon opening terminal

We can now use the 'pwd' *print working directory* command to see what directory or room we are in; the result is returned to us in simple text:

```
pi@raspberrypi:~ $ pwd
/home/pi
pi@raspberrypi:~ $
```

Now we know what directory we are in, but we still don't have any idea what accompanies us; for that we can use 'ls' *list* command; which returns us a list of files and directories in our *working directory*.

```
pi@raspberrypi:~ $ pwd
/home/pi
pi@raspberrypi:~ $ ls
a file.jpg  DHT11_Python  Downloads  Pictures  python_games  Templates
Desktop     Documents     Music      Public    rpi_ws281x    Videos
pi@raspberrypi:~ $
```

I have added a file here to demonstrate that files are coloured differently to directories.

Now we can try entering one of the directories listed, for this we can use the *cd change directory* command. This command takes a 'path' as an argument. Paths are just strings of text describing the location of a directory or file; pwd returns the 'path' of our current working directory. Another path could be '/home/pi/Documents/'. Paths can also be relative to our current location, so if our current working directory is '/home/pi' then 'Documents' would be a valid path.

Most of the time you will use relative paths, they are shorter and easier to work with too.

To navigate to a directory relative to our own we first use ls to see what our options are.

```
pi@raspberrypi:~ $ ls
Desktop  Documents  Music  Public  rpi_ws281x  Videos
DHT11_Python  Downloads  Pictures  python_games  Templates
pi@raspberrypi:~ $
```

Then we use the cd command with a path to enter one, we'll use

the `python_games` directory because we know ahead of time its not empty: `cd python_games`

Another valid and equivalent command would again be `cd /home/pi/python_games`

Once we enter the directory we can use the `ls` command to orient ourselves.

```
pi@raspberrypi:~ $ cd python_games/
pi@raspberrypi:~/python_games $ ls
4row_arrow.png          gem4.png               pentomino.py
4row_black.png          gem5.png               pinkgirl.png
4row_board.png          gem6.png               Plain_Block.png
4row_computerwinner.png gem7.png               princess.png
4row_humanwinner.png   gemgem.py              RedSelector.png
4row_red.png            grass1.png             Rock.png
4row_tie.png            grass2.png             Selector.png
badswap.wav             grass3.png             simulate.py
beep1.ogg               grass4.png             slidepuzzle.py
beep2.ogg               Grass_Block.png        squirrel.png
beep3.ogg               horngirl.png           squirrel.py
beep4.ogg               inkspillogo.png        Star.png
blankpygame.py          inkspill.py            starPusherLevels.txt
boy.png                 inkspillresetbutton.png starpusher.py
catanimation.py         inkspillsettingsbutton.png star_solved.png
catgirl.png             inkspillsettings.png  star_title.png
cat.png                 inkspillspot.png      tetrisb.mid
drawing.py              launcher.sh             tetrisc.mid
flippybackground.png   match0.wav             tetrominoforidiots.py
flippyboard.png        match1.wav             tetromino.py
flippy.py               match2.wav             Tree_Short.png
fourinarow.py           match3.wav             Tree_Tall.png
gameicon.png            match4.wav             Tree_Ugly.png
gem1.png                match5.wav             Wall_Block_Tall.png
gem2.png                memorypuzzle_obfuscated.py Wood_Block_Tall.png
gem3.png                memorypuzzle.py        wormy.py
pi@raspberrypi:~/python_games $ █
```

We can see dozens of files and no directories, here we can practice running and terminating a python script.

## Running and Exiting python scripts

Just like the `cd` command there is a `python` command pre-installed that takes a python script as an argument, python scripts are files with a `.py` extension.

To run a game choose any file with a `.py` extension and use it as an argument to run python.

```

pi@raspberrypi:~/python_games $ ls
4row_arrow.png          gem4.png                pentomino.py
4row_black.png          gem5.png                pinkgirl.png
4row_board.png          gem6.png                Plain_Block.png
4row_computerwinner.png gem7.png                princess.png
4row_humanwinner.png   gengem.py              RedSelector.png
4row_red.png            grass1.png              Rock.png
4row_tie.png            grass2.png              Selector.png
badswap.wav             grass3.png              simulate.py
beep1.ogg               grass4.png              slidepuzzle.py
beep2.ogg               Grass_Block.png         squirrel.png
beep3.ogg               horngirl.png           squirrel.py
beep4.ogg               inkspillogo.png        Star.png
blankpygame.py          inkspill.py             starPusherLevels.txt
boy.png                 inkspillresetbutton.png starpusher.py
catanimation.py         inkspillsettingsbutton.png star_solved.png
catgirl.png             inkspillsettings.png   star_title.png
cat.png                 inkspillspot.png       tetrisb.mid
drawing.py              launcher.sh              tetrisc.mid
flippybackground.png    match0.wav              tetrominoforidiots.py
flippyboard.png         match1.wav              tetromino.py
flippy.py               match2.wav              Tree_Short.png
fourinarow.py           match3.wav              Tree_Tall.png
gameicon.png            match4.wav              Tree_Ugly.png
gem1.png                match5.wav              Wall_Block_Tall.png
gem2.png                memorypuzzle_obfuscated.py Wood_Block_Tall.png
gem3.png                memorypuzzle.py         wormy.py
pi@raspberrypi:~/python_games $ python flippy.py █

```

To exit the program at any time use the **Control-C** keybind, this sends a kill signal to the python process we just started; this will also be used regularly when you want to exit python scripts you are testing.

Next we will navigate back home using terminal commands and create our own directories and files.

Apart from relative and absolute paths that we already covered there are some more special cases, for example the `..` path refers to the directory one level above our current working one. This makes it easy to navigate upwards out of whatever area we entered.

To navigate back to the home directory use `'cd ..'`, and then use `'pwd'` to check if you made it back.

Finally we will create our own directory and some files to populate it. Just like the `cd` command there exists a `mkdir` command which again takes a path. This command simply spawns a folder with our chosen path, for example..

```
pi@raspberrypi:~ $ mkdir my_directory
pi@raspberrypi:~ $ █
```

At first we have no indication that our command did anything, but a simple `ls` will reveal all.

```
pi@raspberrypi:~ $ mkdir my_directory
pi@raspberrypi:~ $ ls
Desktop      Documents  Music      Pictures  python_games  Templates
DHT11_Python Downloads  my_directory Public    rpi_ws281x    Videos
pi@raspberrypi:~ $ █
```

We don't want to spam files all over our home directory, so before we start making them lets enter our directory with `cd`. The command for creating files in the terminal is called *touch*, it takes a path and creates an empty file wherever we tell it too, or if the file already exists it just 'touches' it; updating its last modified date ect.

Now that we are safely in our own directory we can try it out:

```
pi@raspberrypi:~/my_directory $ touch hello
pi@raspberrypi:~/my_directory $ touch world
pi@raspberrypi:~/my_directory $ ls
hello world
pi@raspberrypi:~/my_directory $ █
```

There does exist a `rm` command to *remove* files, but it can be dangerous if you accidentally delete some script or important document you were just working on!

as you can guess it takes a path as its argument; here we can see it at work.

```
pi@raspberrypi:~/my_directory $ rm hello
pi@raspberrypi:~/my_directory $ rm world
pi@raspberrypi:~/my_directory $ ls
pi@raspberrypi:~/my_directory $ █
```

Last of all we can clean up after our empty directory, navigate back to the home using `cd`, and then use the sister to `rm`, `rmdir` to remove our directory.

```
pi@raspberrypi:~$ rmdir my_directory/  
pi@raspberrypi:~$ ls  
Desktop      Documents  Music      Public     rpi_ws281x  Videos  
DHT11_Python Downloads  Pictures   python_games Templates  
pi@raspberrypi:~$ █
```

## Conclusions

In this tutorial we reminded ourselves what directories and files are, how to find our *current working directory* in the terminal, how to list files and folders, how to enter and leave directories, how to run and exit python scripts, and how to create and delete files / folders.