February 2022 Update

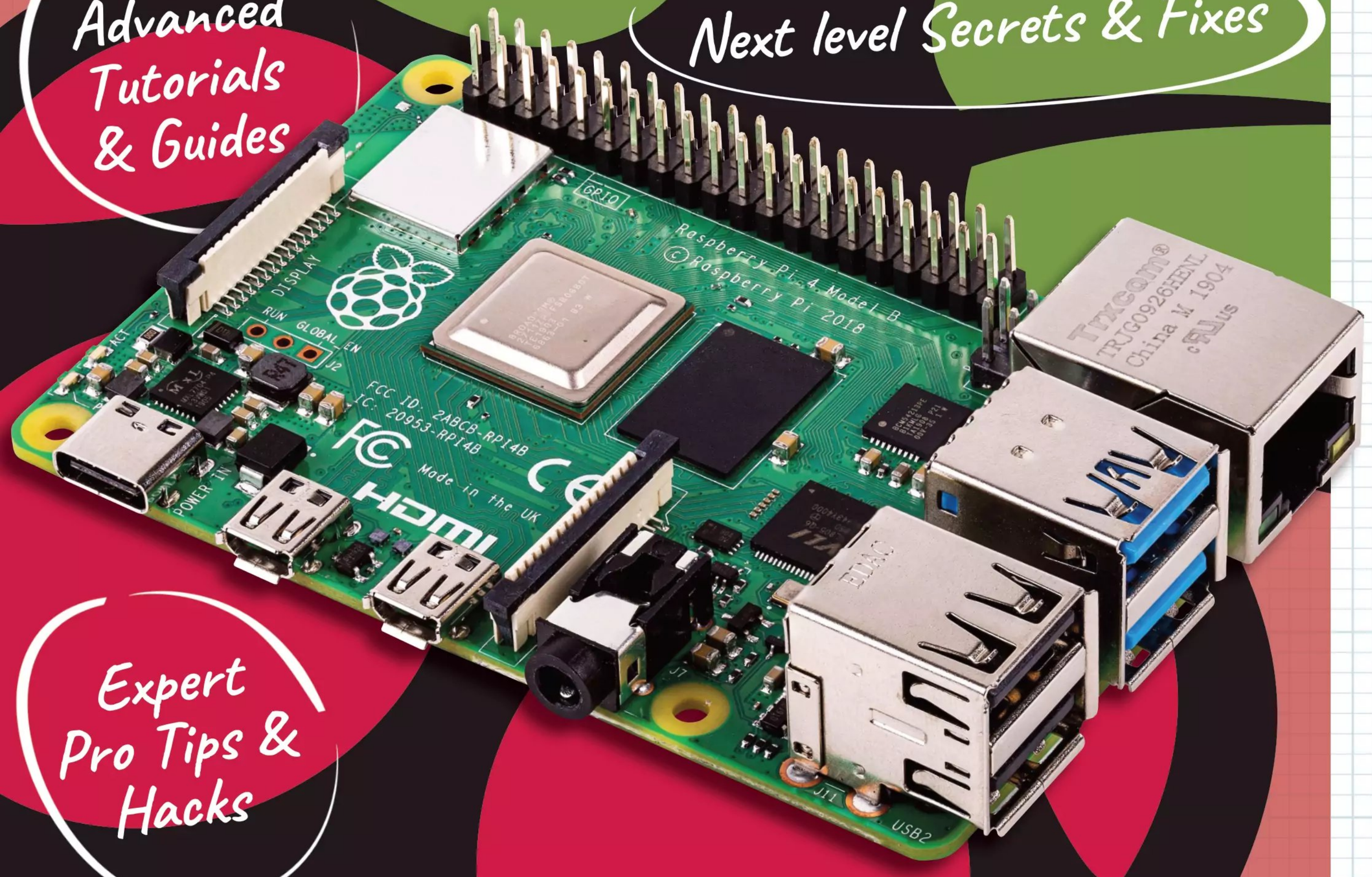
Raspberry Pi

Tricks and Tips

Raspberry Pi 4 • Raspberry Pi PICO • Raspberry Pi 400

Advanced Tutorials & Guides

Next level Secrets & Fixes



Expert Pro Tips & Hacks

Over 467 Secrets & Hacks

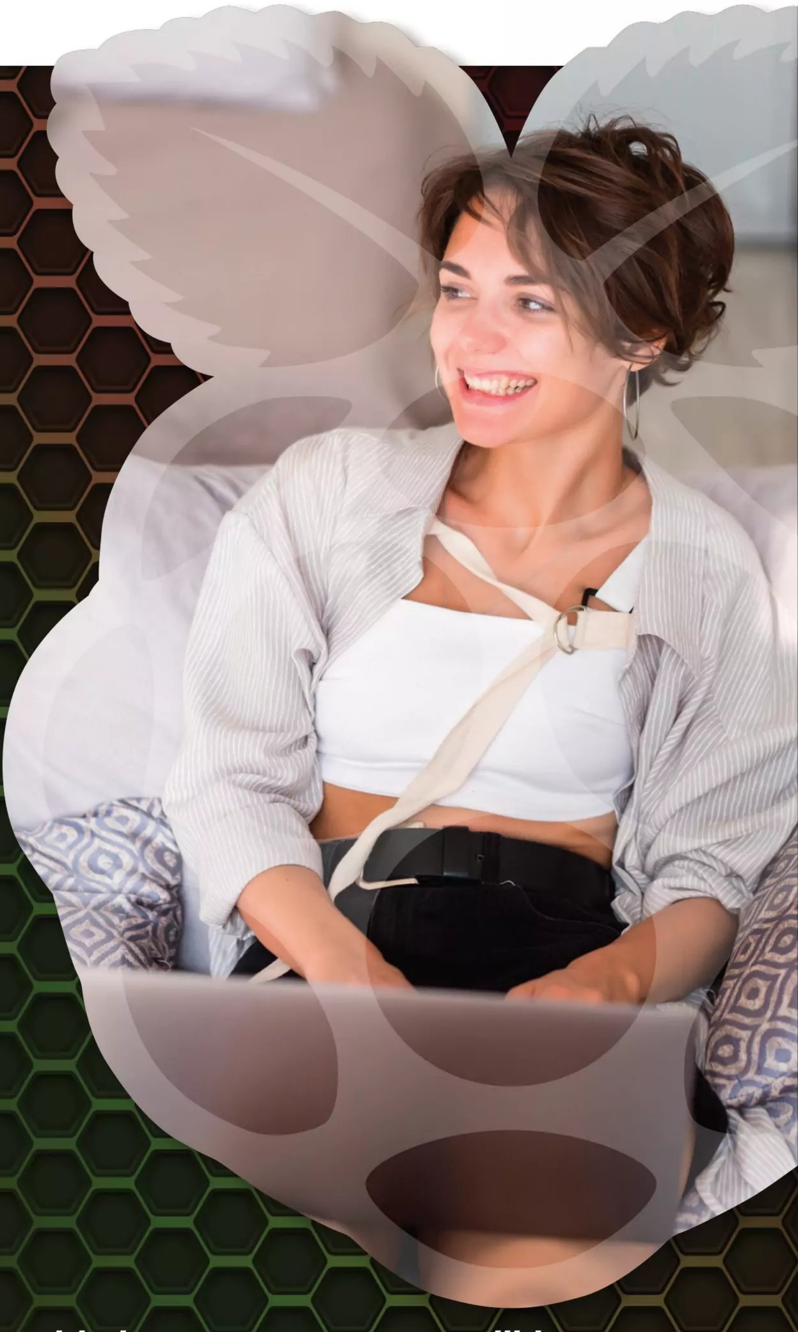
Raspberry Pi
Tricks and Tips

Welcome to the next level of understanding and expansion of your user experience!

For some it is enough to master the basics of your new device, software or hobby. Yet for many, like you, that's just the start! Advancing your skill set is the goal of all users of consumer technology and our team of long term industry experts will help you achieve exactly that. Over this extensive series of titles we will be looking in greater depth at how you make the absolute most from the latest consumer electronics, software, hobbies and trends! We will guide you step-by-step through using all the advanced aspects of the technology that you may have been previously apprehensive at attempting. Let our expert guide help you build your understanding of technology and gain the skills to take you from a confident user to an experienced expert.

Over the page our journey continues, and we will be with you at every stage to advise, inform and ultimately inspire you to go further.

Contents



60 Pi Projects

"With our Raspberry Pi guide in your corner, you will learn and discover how the Pi works, what you can do with it and where to take your Pi adventures. We've got everything from setting up the Pi to learning how to code on it using Python, and from learning Linux to projects that'll take you and your Pi to the next level.

Read on, and let's unleash your imagination with the power of the Raspberry Pi"

Python Code and Ideas

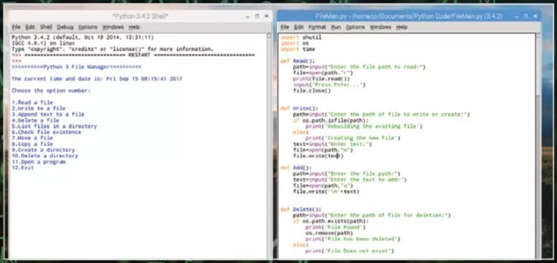
To help you get the most from your Pi and Python, we've included some type-in code that you can use in your own programming projects. Use the code, take it apart, improve it and see what amazing things you can create through Python and the Raspberry Pi.

This code repository is available from https://bdmpublications.com/code-portal/, and features all the listings in these pages and more. Here, you'll find code for games, scrolling animations and even a file manager.

This is a great resource, so sign up, get the code and let us know what creative coding content you've developed with your Raspberry Pi.

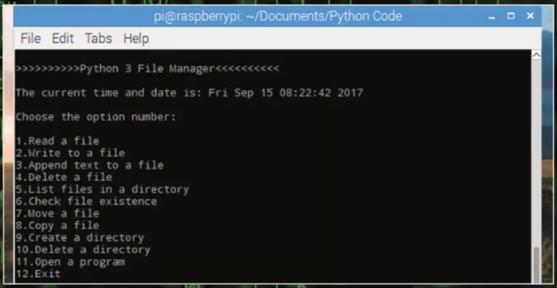
Python File Manager

This file manager program displays a list of options that allow you to read a file, write to a file, append to a file, delete a file, list the contents of a directory and much more. It's remarkably easy to edit and insert into your own code, or add to.


Python 3.4.2 Shell
File Edit Shell Debug Options Windows Help
FileMan.py - /home/pi/Documents/Python Code/FileMan.py (3.4.2)
File Edit Format Run Options Windows Help

```
Python 3.4.2 (default, Oct 19 2014, 13:31:11)
[GCC 4.9.1] on linux
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
>>>>>>>>>>Python 3 File Manager<<<<<<<<<<

The current time and date is: Fri Sep 15 08:15:41 2017

Choose the option number:

1.Read a file
2.Write to a file
3.Append text to a file
4.Delete a file
5.List files in a directory
6.Check file existence
7.Move a file
8.Copy a file
9.Create a directory
10.Delete a directory
11.Open a program
12.Exit
```

```python
import shutil
import os
import time

def Read():
    path=input("Enter the file path to read:")
    file=open(path,"r")
    print(file.read())
    input('Press Enter...')
    file.close()


def Write():
    path=input("Enter the path of file to write or create:")
    if os.path.isfile(path):
        print('Rebuilding the existing file')
    else:
        print('Creating the new file')
    text=input("Enter text:")
    file=open(path,"w")
    file.write(text)

def Add():
    path=input("Enter the file path:")
    text=input("Enter the text to add:")
    file=open(path,"a")
    file.write('\n'+text)


def Delete():
    path=input("Enter the path of file for deletion:")
    if os.path.exists(path):
        print('File Found')
        os.remove(path)
        print('File has been deleted')
    else:
        print('File Does not exist')
```



```
>>>>>>>>>>Python 3 File Manager<<<<<<<<<<

The current time and date is: Fri Sep 15 08:22:42 2017

Choose the option number:

1.Read a file
2.Write to a file
3.Append text to a file
4.Delete a file
5.List files in a directory
6.Check file existence
7.Move a file
8.Copy a file
9.Create a directory
10.Delete a directory
11.Open a program
12.Exit
```

1 This part of the code imports the necessary modules. The OS and Subprocess modules deal with the operating system elements of the program.

2 Each def XXX() functions store the code for each of the menu's options. Once the code within the function is complete, the code returns to the main menu for another option.

3 This is part of the code that checks to see what OS the user is running. In Windows the CLS command clears the screen, whereas in Linux and macOS, the Clear command wipes the screen. If the code tries to run CLS when being used in Linux or macOS, an error occurs, which then prompts it to run the Clear command instead.

4 These are the options, from 1 to 12. Each executes the appropriate function when the relevant number is entered.

FILEMAN.PY

Copy the code below into a New > File and save it as FileMan.py. Once executed it will display the program title, along with the current time and date and the available options.

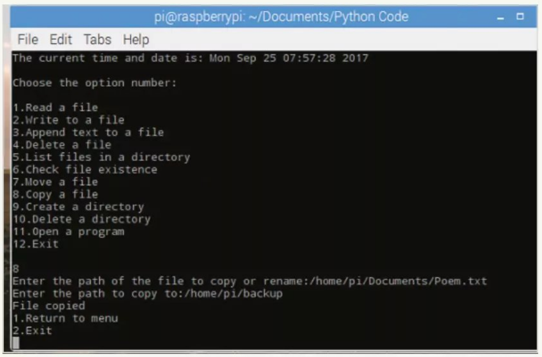
```
import shutil          ┐
import os              │ 1
import time            │
import subprocess      ┘

def Read():
   path=input("Enter the file path to read:")   ┐
   file=open(path,"r")                          │
   print(file.read())                           │ 2
   input('Press Enter...')                      │
   file.close()                                 ┘

def Write():
   path=input("Enter the path of file to write or create:")
   if os.path.isfile(path):
      print('Rebuilding the existing file')
   else:
      print('Creating the new file')
   text=input("Enter text:")
   file=open(path,"w")
   file.write(text)

def Add():
   path=input("Enter the file path:")
   text=input("Enter the text to add:")
   file=open(path,"a")
   file.write('\n'+text)

def Delete():
   path=input("Enter the path of file for deletion:")
   if os.path.exists(path):
      print('File Found')
      os.remove(path)
      print('File has been deleted')
   else:
      print('File Does not exist')

def Dirlist():
   path=input("Enter the Directory path to display:")
   sortlist=sorted(os.listdir(path))
   i=0
   while(i<len(sortlist)):
      print(sortlist[i]+'\n')
      i+=1

def Check():
   fp=int(input('Check existence of \n1.File \n2.
   Directory\n'))
      if fp==1:
         path=input("Enter the file path:")
         os.path.isfile(path)
```

```
        if os.path.isfile(path)==True:
            print(‘File Found’)
        else:
            print(‘File not found’)
    if fp==2:
        path=input(“Enter the directory path:”)
        os.path.isdir(path)
        if os.path.isdir(path)==False:
            print(‘Directory Found’)
        else:
            print(‘Directory Not Found’)

def Move():
    path1=input(‘Enter the source path of file to move:’)
    mr=int(input(‘1.Rename \n2.Move \n’))
    if mr==1:
        path2=input(‘Enter the destination path and file name:’)
        shutil.move(path1,path2)
        print(‘File renamed’)
    if mr==2:
        path2=input(‘Enter the path to move:’)
        shutil.move(path1,path2)
        print(‘File moved’)

def Copy():
    path1=input(‘Enter the path of the file to copy or rename:’)
    path2=input(‘Enter the path to copy to:’)
    shutil.copy(path1,path2)
    print(‘File copied’)

def Makedir():
    path=input(“Enter the directory name with path to make
    \neg. C:\\Hello\\Newdir \nWhere Newdir is new
    directory:”)
    os.makedirs(path)
    print(‘Directory Created’)

def Removedir():
    path=input(‘Enter the path of Directory:’)
    treedir=int(input(‘1.Deleted Directory \n2.Delete
    Directory Tree \n3.Exit \n’))
    if treedir==1:
        os.rmdir(path)
    if treedir==2:
        shutil.rmtree(path)
        print(‘Directory Deleted’)
    if treedir==3:
        exit()

def Openfile():
    path=input(‘Enter the path of program:’)
    try:
        os.startfile(path)
    except:
        print(‘File not found’)

run=1
while(run==1):
    try:
        os.system(‘clear’)
    except OSError:
        os.system(‘cls’)
```

3

```
    print(‘\n>>>>>>>>>>Python 3 File Manager<<<<<<<<<<\n’)
    print(‘The current time and date is:’,time.asctime())
    print(‘\nChoose the option number: \n’)
    dec=int(input(‘’’1.Read a file
2.Write to a file
3.Append text to a file
4.Delete a file
5.List files in a directory
6.Check file existence
7.Move a file
8.Copy a file
9.Create a directory
10.Delete a directory
11.Open a program
12.Exit

‘’’))
    if dec==1:
        Read()
    if dec==2:
        Write()
    if dec==3:
        Add()
    if dec==4:
        Delete()
    if dec==5:
        Dirlist()
    if dec==6:
        Check()
    if dec==7:
        Move()
    if dec==8:
        Copy()
    if dec==9:
        Makedir()
    if dec==10:
        Removedir()
    if dec==11:
        Openfile()
    if dec==12:
        exit()
    run=int(input(“1.Return to menu\n2.Exit \n”))
    if run==2:
        exit()
```

4



```
The current time and date is: Mon Sep 25 07:57:28 2017

Choose the option number:

1.Read a file
2.Write to a file
3.Append text to a file
4.Delete a file
5.List files in a directory
6.Check file existence
7.Move a file
8.Copy a file
9.Create a directory
10.Delete a directory
11.Open a program
12.Exit

8
Enter the path of the file to copy or rename:/home/pi/Documents/Poem.txt
Enter the path to copy to:/home/pi/backup
File copied
1.Return to menu
2.Exit
```

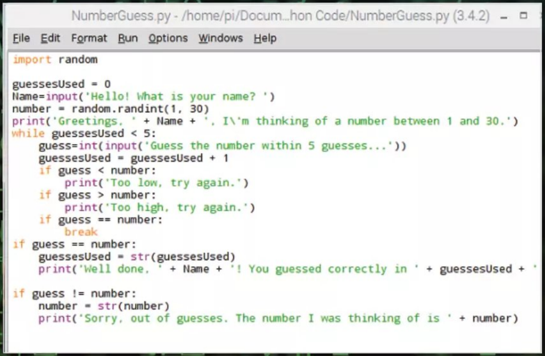
Imports

There are three modules to import here: Shutil, OS and Time. The first two deal with the operating system and file management and manipulation; and the Time module simply displays the current time and date.

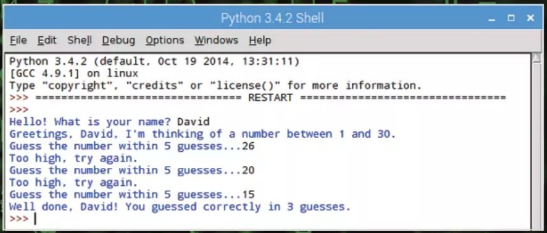
Note how we've included a try and except block to check if the user is running the code on a Linux system or Windows. Windows uses CLS to clear the screen, while Linux uses clear. The try block should work well enough but it's a point of possible improvement depending on your own system.

Number Guessing Game

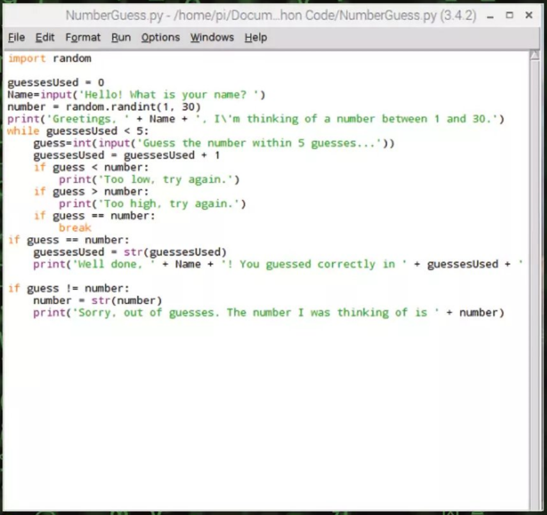
This is a simple little piece of code but it makes good use of the Random module, print and input, and a while loop. The number of guesses can be increased from 5 and the random number range can easily be altered too.


NumberGuess.py - /home/pi/Docum...hon Code/NumberGuess.py (3.4.2)
File Edit Format Run Options Windows Help

```
import random

guessesUsed = 0
Name=input('Hello! What is your name? ')
number = random.randint(1, 30)
print('Greetings, ' + Name + ', I\'m thinking of a number between 1 and 30.')
while guessesUsed < 5:
    guess=int(input('Guess the number within 5 guesses...'))
    guessesUsed = guessesUsed + 1
    if guess < number:
        print('Too low, try again.')
    if guess > number:
        print('Too high, try again.')
    if guess == number:
        break
if guess == number:
    guessesUsed = str(guessesUsed)
    print('Well done, ' + Name + '! You guessed correctly in ' + guessesUsed + '

if guess != number:
    number = str(number)
    print('Sorry, out of guesses. The number I was thinking of is ' + number)
```



```
Python 3.4.2 (default, Oct 19 2014, 13:31:11)
[GCC 4.9.1] on linux
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
Hello! What is your name? David
Greetings, David, I'm thinking of a number between 1 and 30.
Guess the number within 5 guesses...26
Too high, try again.
Guess the number within 5 guesses...20
Too high, try again.
Guess the number within 5 guesses...15
Well done, David! You guessed correctly in 3 guesses.
>>> |
```


NumberGuess.py - /home/pi/Docum...hon Code/NumberGuess.py (3.4.2)
File Edit Format Run Options Windows Help

```
import random

guessesUsed = 0
Name=input('Hello! What is your name? ')
number = random.randint(1, 30)
print('Greetings, ' + Name + ', I\'m thinking of a number between 1 and 30.')
while guessesUsed < 5:
    guess=int(input('Guess the number within 5 guesses...'))
    guessesUsed = guessesUsed + 1
    if guess < number:
        print('Too low, try again.')
    if guess > number:
        print('Too high, try again.')
    if guess == number:
        break
if guess == number:
    guessesUsed = str(guessesUsed)
    print('Well done, ' + Name + '! You guessed correctly in ' + guessesUsed + '

if guess != number:
    number = str(number)
    print('Sorry, out of guesses. The number I was thinking of is ' + number)
```

NUMBERGUESS.PY

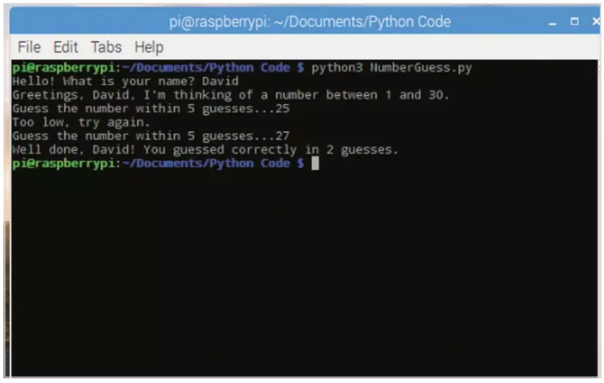
Copy the code and see if you can beat the computer within five guesses. It's an interesting bit of code that can be quite handy when your implementing a combination of the Random module alongside a while loop.

```
import random                                                          1

guessesUsed = 0
Name=input(‘Hello! What is your name? ’)
number = random.randint(1, 30)
print(‘Greetings, ’ + Name + ‘, I\’m thinking of a
number between 1 and 30.’)
while guessesUsed < 5:
    guess=int(input(‘Guess the number within 5 guesses...’))
    guessesUsed = guessesUsed + 1
    if guess < number:
        print(‘Too low, try again.’)                                   2
    if guess > number:
        print(‘Too high, try again.’)
    if guess == number:
        break
if guess == number:
    guessesUsed = str(guessesUsed)
    print(‘Well done, ’ + Name + ‘! You guessed
    correctly in ’ + guessesUsed + ‘ guesses.’)
                                                                       3
if guess != number:
    number = str(number)
    print(‘Sorry, out of guesses. The number I was
    thinking of is ’ + number)
```

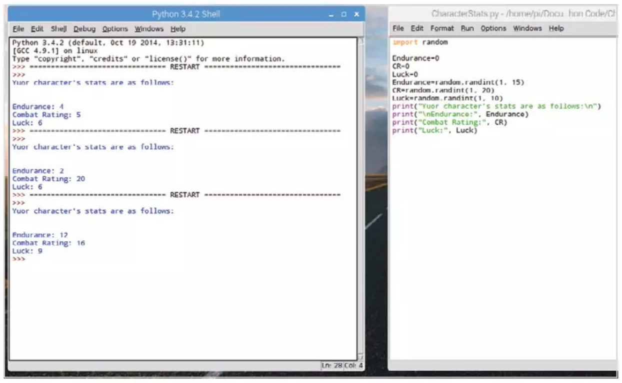
1 Although this is a reasonably easy to follow program, there are some elements to the code that are worth pointing out. To begin with, you need to import the Random module, as you're using random numbers within the code.

2 This section of the code creates the variables for the number of guesses used, along with the name of the player, and also sets up the random number between 1 and 30. If you want a wider range of random number selection, then increase the **number=random.randint(1, 30)** end value of 30; don't make it too high though or the player will never be able to guess it. If the player guesses too low or too high, they are given the appropriate output and asked to try again, while the number of guesses is less than five. You can also increase the number of guesses from 5 by altering the **while guessesUsed < 5:** value.

3 If the player guessed the correct number then they are given a 'well done' output, along with how many guesses they used up. If the player runs out of guesses, then the game over output is displayed instead, along with revealing the number the computer was thinking of. Remember, if you do alter the values of the random number chosen by the computer, or the number of guesses the player can take, then along with the variable values, you also need to amend the instructions given in the print statements at the start of the code.



```
pi@raspberrypi:~/Documents/Python Code $ python3 NumberGuess.py
Hello! What is your name? David
Greetings, David. I'm thinking of a number between 1 and 30.
Guess the number within 5 guesses...25
Too low, try again.
Guess the number within 5 guesses...27
Well done, David! You guessed correctly in 2 guesses.
pi@raspberrypi:~/Documents/Python Code $
```



```
Python 3.4.2 (default, Oct 19 2014, 13:31:11)
[GCC 4.9.1] on linux
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
Yuor character's stats are as follows:


Endurance: 4
Combat Rating: 5
Luck: 6
>>> ================================ RESTART ================================
>>>
Yuor character's stats are as follows:


Endurance: 2
Combat Rating: 20
Luck: 6
>>> ================================ RESTART ================================
>>>
Yuor character's stats are as follows:


Endurance: 12
Combat Rating: 16
Luck: 9
>>>
```

```
import random

Endurance=0
CR=0
Luck=0
Endurance=random.randint(1, 15)
CR=random.randint(1, 20)
Luck=random.randint(1, 10)
print("Yuor character's stats are as follows:\n")
print("\nEndurance:", Endurance)
print("Combat Rating:", CR)
print("Luck:", Luck)
```

Code Improvements

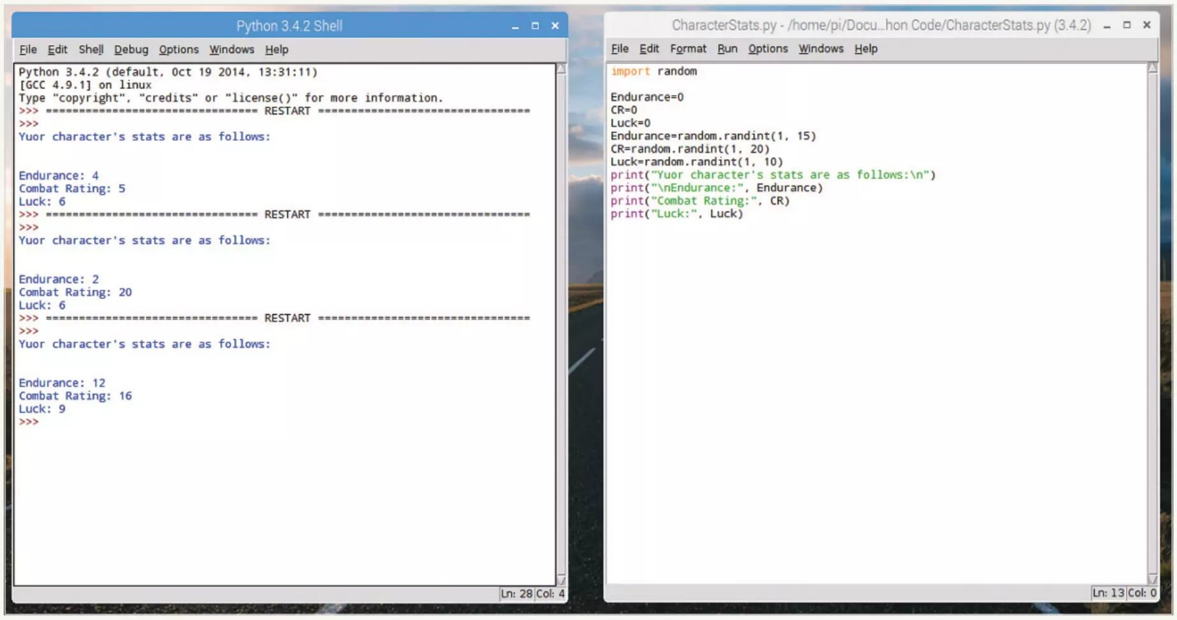
Since this is such as simple script to apply to a situation, there's plenty of room to mess around with it and make it more interesting. Perhaps you can include an option to take score, the best out of three rounds. Maybe an elaborate way to congratulate the player for getting a 'hole in one' correct guess on their first try.

Moreover, the number guessing game code does offer some room for implementing into your code in a different manner. What we mean by this is, the code can be used to retrieve a random number between a range, which in turn can give you the start of a character creation defined function within an adventure game.

Imagine the start of a text adventure written in Python, where the player names their character. The next step is to roll the virtual random dice to decide what that character's combat rating, strength, endurance and luck values are. These can then be carried forward into the game under a set of variables that can be reduced or increased depending on the circumstances the player's character ends up in.
For example, as per the screenshot provided, you could use something along the lines of:

```
Endurance=0
CR=0
Luck=0
Endurance = random.randint(1, 15)
CR = random.randint(1, 20)
Luck = random.randint(1, 10)
Print("Your character's stats are as follows:\n")
Print("Endurance:", Endurance)
Print("Combat Rating:", CR)
Print("Luck:", Luck)
```

The player can then decide to either stick with their roll or try again for the hope of better values being picked. There's ample ways in which to implement this code into a basic adventure game.



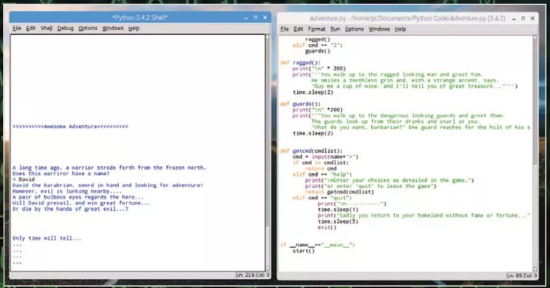
Python 3.4.2 Shell
File Edit Shell Debug Options Windows Help
Ln: 28 Col: 4
CharacterStats.py - /home/pi/Docu...hon Code/CharacterStats.py (3.4.2)
File Edit Format Run Options Windows Help
Ln: 13 Col: 0

```
Python 3.4.2 (default, Oct 19 2014, 13:31:11)
[GCC 4.9.1] on linux
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
Yuor character's stats are as follows:


Endurance: 4
Combat Rating: 5
Luck: 6
>>> ================================ RESTART ================================
>>>
Yuor character's stats are as follows:


Endurance: 2
Combat Rating: 20
Luck: 6
>>> ================================ RESTART ================================
>>>
Yuor character's stats are as follows:


Endurance: 12
Combat Rating: 16
Luck: 9
>>>
```

```
import random

Endurance=0
CR=0
Luck=0
Endurance=random.randint(1, 15)
CR=random.randint(1, 20)
Luck=random.randint(1, 10)
print("Yuor character's stats are as follows:\n")
print("\nEndurance:", Endurance)
print("Combat Rating:", CR)
print("Luck:", Luck)
```

Text Adventure Script

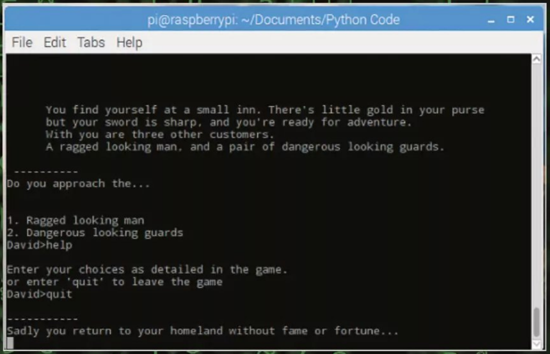
Text adventures are an excellent way to build your Python coding skills and have some fun at the same time. This example that we created will start you on the path to making a classic text adventure; where it will end is up to you.


Python 3.4.2 Shell
File Edit Shell Debug Options Windows Help
Ln: 218 Col: 0

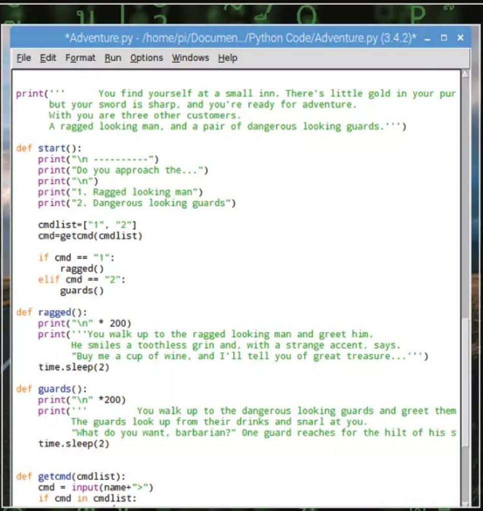
```
>>>>>>>>>>Awesome Adventure<<<<<<<<<<



A long time ago, a warrior strode forth from the frozen north.
Does this warrior have a name?
> David
David the barabrian, sword in hand and looking for adventure!
However, evil is lurking nearby....
A pair of bulbous eyes regards the hero...
Will David prevail, and win great fortune...
Or die by the hands of great evil...?



Only time will tell...
...
...
...
...
```

Adventure.py - /home/pi/Documents/Python Code/Adventure.py (3.4.2)
File Edit Format Run Options Windows Help
Ln: 89 Col: 0

```
        ragged()
    elif cmd == "2":
        guards()

def ragged():
    print("\n" * 200)
    print('''You walk up to the ragged looking man and greet him.
        He smiles a toothless grin and, with a strange accent, says.
        "Buy me a cup of wine, and I'll tell you of great treasure...''')
    time.sleep(2)

def guards():
    print("\n" *200)
    print('''You walk up to the dangerous looking guards and greet them.
        The guards look up from their drinks and snarl at you.
        "What do you want, barbarian?" One guard reaches for the hilt of his s
    time.sleep(2)


def getcmd(cmdlist):
    cmd = input(name+">")
    if cmd in cmdlist:
        return cmd
    elif cmd == "help":
        print("\nEnter your choices as detailed in the game.")
        print("or enter 'quit' to leave the game")
        return getcmd(cmdlist)
    elif cmd == "quit":
            print("\n----------")
            time.sleep(1)
            print("Sadly you return to your homeland without fame or fortune..."
            time.sleep(5)
            exit()


if __name__=="__main__":
    start()
```



```
     You find yourself at a small inn. There's little gold in your purse
     but your sword is sharp, and you're ready for adventure.
     With you are three other customers.
     A ragged looking man, and a pair of dangerous looking guards.

 ----------
Do you approach the...


1. Ragged looking man
2. Dangerous looking guards
David>help

Enter your choices as detailed in the game.
or enter 'quit' to leave the game
David>quit

----------
Sadly you return to your homeland without fame or fortune...
```


Adventure.py - /home/pi/Documen.../Python Code/Adventure.py (3.4.2)
File Edit Format Run Options Windows Help

```
print('''      You find yourself at a small inn. There's little gold in your pur
     but your sword is sharp, and you're ready for adventure.
     With you are three other customers.
     A ragged looking man, and a pair of dangerous looking guards.''')

def start():
    print("\n ----------")
    print("Do you approach the...")
    print("\n")
    print("1. Ragged looking man")
    print("2. Dangerous looking guards")

    cmdlist=["1", "2"]
    cmd=getcmd(cmdlist)

    if cmd == "1":
        ragged()
    elif cmd == "2":
        guards()

def ragged():
    print("\n" * 200)
    print('''You walk up to the ragged looking man and greet him.
        He smiles a toothless grin and, with a strange accent, says.
        "Buy me a cup of wine, and I'll tell you of great treasure...''')
    time.sleep(2)

def guards():
    print("\n" *200)
    print('''        You walk up to the dangerous looking guards and greet them
        The guards look up from their drinks and snarl at you.
        "What do you want, barbarian?" One guard reaches for the hilt of his s
    time.sleep(2)


def getcmd(cmdlist):
    cmd = input(name+">")
    if cmd in cmdlist:
```

ADVENTURE.PY

The Adventure game uses just the Time module to begin with, creating pauses between print functions. There's a help system in place to expand upon, as well as the story itself.

```
import time

print("\n" * 200)
print(">>>>>>>>>>Awesome Adventure<<<<<<<<<<\n")
print("\n" * 3)
time.sleep(3)
print("\nA long time ago, a warrior strode forth from
the frozen north.")
time.sleep(1)
print("Does this warrior have a name?")
name=input("> ")
print(name, "the barbarian, sword in hand and looking
for adventure!")
time.sleep(1)
print("However, evil is lurking nearby....")
time.sleep(1)
print("A pair of bulbous eyes regards the hero...")
time.sleep(1)
print("Will", name, "prevail, and win great fortune...")
time.sleep(1)
print("Or die by the hands of great evil...?")
time.sleep(1)
print("\n" *3)
print("Only time will tell...")
time.sleep(1)
print('...')
time.sleep(1)
print('...')
time.sleep(1)
print('...')
time.sleep(1)
print('...')
time.sleep(5)
print("\n" *200)

print('''      You find yourself at a small inn. There's
   little gold in your purse but your sword is sharp,
   and you're ready for adventure.
   With you are three other customers.
   A ragged looking man, and a pair of dangerous
   looking guards.''')

def start():
   print("\n ----------")
   print("Do you approach the...")
   print("\n")
   print("1. Ragged looking man")
   print("2. Dangerous looking guards")

   cmdlist=["1", "2"]
   cmd=getcmd(cmdlist)
```

```
  if cmd == "1":
     ragged()
  elif cmd == "2":
     guards()

def ragged():
  print("\n" * 200)
  print('''You walk up to the ragged looking man and
  greet him.
     He smiles a toothless grin and, with a strange
     accent, says.
     "Buy me a cup of wine, and I'll tell you of
     great treasure...''')
  time.sleep(2)

def guards():
  print("\n" *200)
  print('''You walk up to the dangerous looking guards
  and greet them.
     The guards look up from their drinks and
     snarl at you.
     "What do you want, barbarian?" One guard reaches
     for the hilt of his sword...''')
  time.sleep(2)
```

```
def getcmd(cmdlist):
  cmd = input(name+">")
  if cmd in cmdlist:
     return cmd
  elif cmd == "help":
     print("\nEnter your choices as detailed in
     the game.")
     print("or enter 'quit' to leave the game")
     return getcmd(cmdlist)
  elif cmd == "quit":
     print("\n-----------")
     time.sleep(1)
     print("Sadly you return to your homeland without
     fame or fortune...")
     time.sleep(5)
     exit()


if _ _name_ _ ==" _ _main_ _":
  start()
```

Adventure Time

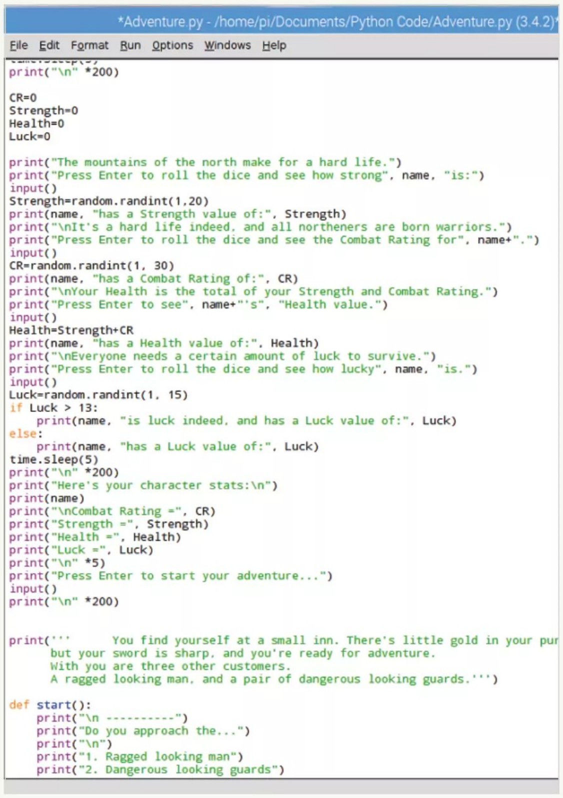
This, as you can see, is just the beginning of the adventure and takes up a fair few lines of code. When you expand it, and weave the story along, you'll find that you can repeat certain instances such as a chance meeting with an enemy or the like.

We've created each of the two encounters as a defined set of functions, along with a list of possible choices under the cmdlist list, and cmd variable, of which is also a defined function. Expanding on this is quite easy, just map out each encounter and choice and create a defined function around it. Providing the user doesn't enter quit into the adventure, they can keep playing.

There's also room in the adventure for a set of variables designed for combat, luck, health, endurance and even an inventory or amount of gold earned. Each successful combat situation can reduce the main character's health but increase their combat skills or endurance. Plus, they could loot the body and gain gold, or earn gold through quests.

Finally, how about introducing the Random module. This will enable you to include an element of chance in the game. For example, in combat, when you strike an enemy you will do a random amount of damage as will they. You could even work out the maths behind improving the chance of a better hit based on your or your opponent's combat skills, current health, strength and endurance. You could create a game of dice in the inn, to see if you win or lose gold (again, improve the chances of winning by working out your luck factor into the equation).

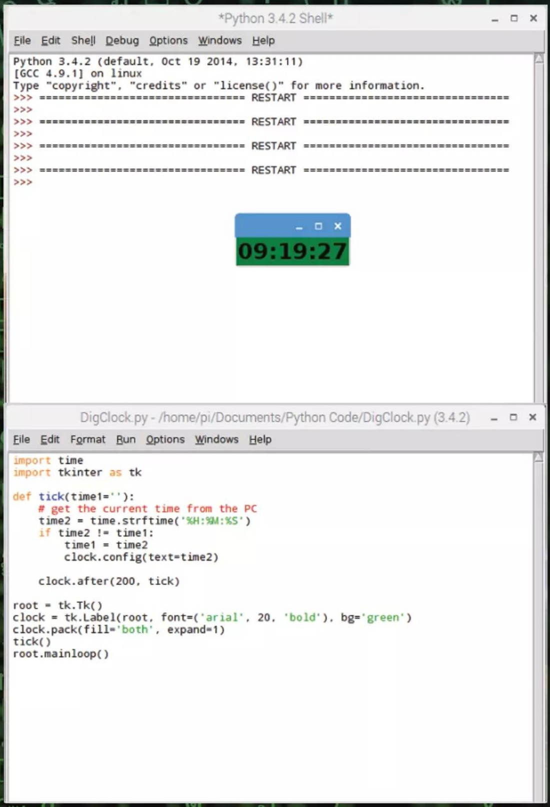
Needless to say, your text adventure can grow exponentially and prove to be a work of wonder. Good luck, and have fun with your adventure.


Adventure.py - /home/pi/Documents/Python Code/Adventure.py (3.4.2)
File Edit Format Run Options Windows Help

```
print("\n" *200)

CR=0
Strength=0
Health=0
Luck=0

print("The mountains of the north make for a hard life.")
print("Press Enter to roll the dice and see how strong", name, "is:")
input()
Strength=random.randint(1,20)
print(name, "has a Strength value of:", Strength)
print("\nIt's a hard life indeed. and all northeners are born warriors.")
print("Press Enter to roll the dice and see the Combat Rating for", name+".")
input()
CR=random.randint(1, 30)
print(name, "has a Combat Rating of:", CR)
print("\nYour Health is the total of your Strength and Combat Rating.")
print("Press Enter to see", name+"'s", "Health value.")
input()
Health=Strength+CR
print(name, "has a Health value of:", Health)
print("\nEveryone needs a certain amount of luck to survive.")
print("Press Enter to roll the dice and see how lucky", name, "is.")
input()
Luck=random.randint(1, 15)
if Luck > 13:
    print(name, "is luck indeed, and has a Luck value of:", Luck)
else:
    print(name, "has a Luck value of:", Luck)
time.sleep(5)
print("\n" *200)
print("Here's your character stats:\n")
print(name)
print("\nCombat Rating =", CR)
print("Strength =", Strength)
print("Health =", Health)
print("Luck =", Luck)
print("\n" *5)
print("Press Enter to start your adventure...")
input()
print("\n" *200)


print('''      You find yourself at a small inn. There's little gold in your pur
      but your sword is sharp, and you're ready for adventure.
      With you are three other customers.
      A ragged looking man, and a pair of dangerous looking guards.''')

def start():
    print("\n ----------")
    print("Do you approach the...")
    print("\n")
    print("1. Ragged looking man")
    print("2. Dangerous looking guards")
```

Python Digital Clock

There is already a clock displayed on the desktop of most operating systems but it's always handy to have one on top of the currently open window. To that end, why not create a Python digital clock that can be a companion desktop widget for you.


Python 3.4.2 Shell
File Edit Shell Debug Options Windows Help
09:19:27
DigClock.py - /home/pi/Documents/Python Code/DigClock.py (3.4.2)
File Edit Format Run Options Windows Help

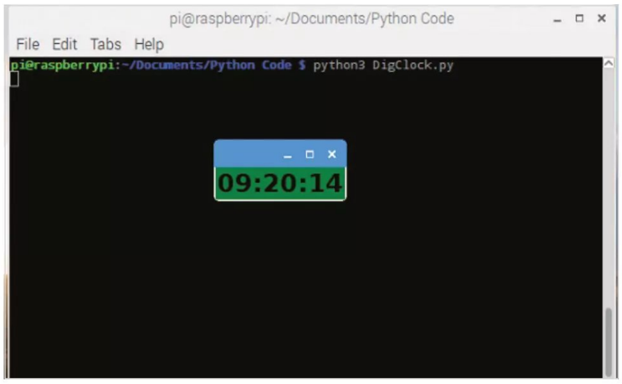
```
Python 3.4.2 (default, Oct 19 2014, 13:31:11)
[GCC 4.9.1] on linux
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
>>> ================================ RESTART ================================
>>>
>>> ================================ RESTART ================================
>>>
>>> ================================ RESTART ================================
>>>
```

```
import time
import tkinter as tk

def tick(time1=''):
    # get the current time from the PC
    time2 = time.strftime('%H:%M:%S')
    if time2 != time1:
        time1 = time2
        clock.config(text=time2)

    clock.after(200, tick)

root = tk.Tk()
clock = tk.Label(root, font=('arial', 20, 'bold'), bg='green')
clock.pack(fill='both', expand=1)
tick()
root.mainloop()
```

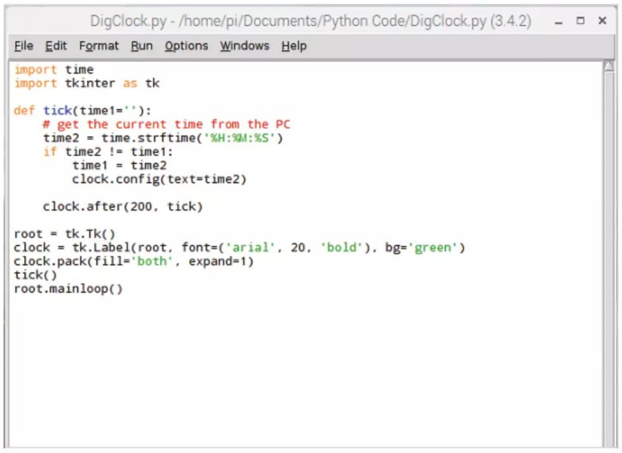
DIGCLOCK.PY

This is a surprisingly handy little script and one that we've used in the past instead of relying on a watch or even the clock in the system tray of the operating system.

```
import time
import tkinter as tk

def tick(time1=''):
   # get the current time from the PC
   time2 = time.strftime('%H:%M:%S')
   if time2 != time1:
      time1 = time2
      clock.config(text=time2)

   clock.after(200, tick)

root = tk.Tk()
clock = tk.Label(root, font=('arial', 20, 'bold'),
bg='green')
clock.pack(fill='both', expand=1)
tick()
root.mainloop()
```



```
pi@raspberrypi:~/Documents/Python Code $ python3 DigClock.py
```


DigClock.py - /home/pi/Documents/Python Code/DigClock.py (3.4.2)
File Edit Format Run Options Windows Help

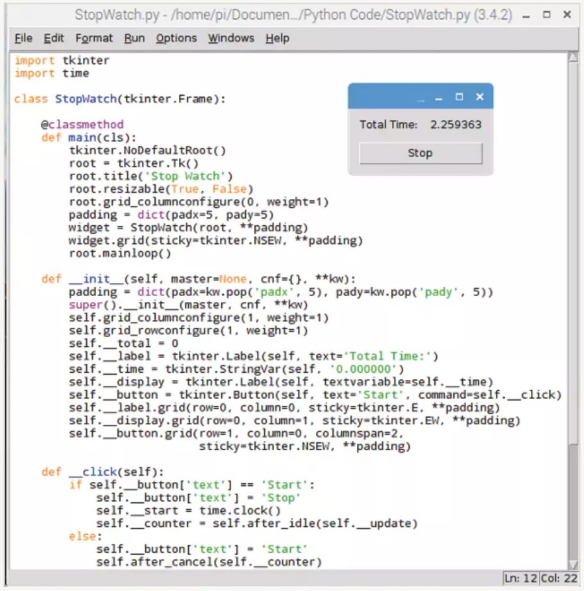
```
import time
import tkinter as tk

def tick(time1=''):
    # get the current time from the PC
    time2 = time.strftime('%H:%M:%S')
    if time2 != time1:
        time1 = time2
        clock.config(text=time2)

    clock.after(200, tick)

root = tk.Tk()
clock = tk.Label(root, font=('arial', 20, 'bold'), bg='green')
clock.pack(fill='both', expand=1)
tick()
root.mainloop()
```

Tick Tock

This is a piece of code we've used many times in the past to keep track of time while working on multiple monitors and with just a quick glance to where we've placed it on the screen.

The Tkinter box can be moved around without affecting the time, maximised or closed by the user at will. We haven't given the Tkinter clock window a title, so you can add to that easily enough by snipping the code from other examples in this book.

Another area of improvement is to include this code when Windows or Linux starts, so it automatically pops up on the desktop. See also, if you're able to improve its functionality by including different time zones: Rome, Paris, London, New York, Moscow and so on.


StopWatch.py - /home/pi/Documen.../Python Code/StopWatch.py (3.4.2)
File Edit Format Run Options Windows Help
Total Time: 2.259363
Stop
Ln: 12 Col: 22

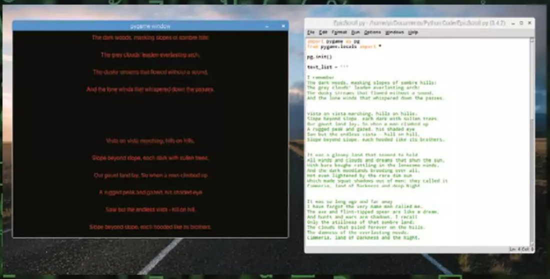
```
import tkinter
import time

class StopWatch(tkinter.Frame):

    @classmethod
    def main(cls):
        tkinter.NoDefaultRoot()
        root = tkinter.Tk()
        root.title('Stop Watch')
        root.resizable(True, False)
        root.grid_columnconfigure(0, weight=1)
        padding = dict(padx=5, pady=5)
        widget = StopWatch(root, **padding)
        widget.grid(sticky=tkinter.NSEW, **padding)
        root.mainloop()

    def __init__(self, master=None, cnf={}, **kw):
        padding = dict(padx=kw.pop('padx', 5), pady=kw.pop('pady', 5))
        super().__init__(master, cnf, **kw)
        self.grid_columnconfigure(1, weight=1)
        self.grid_rowconfigure(1, weight=1)
        self.__total = 0
        self.__label = tkinter.Label(self, text='Total Time:')
        self.__time = tkinter.StringVar(self, '0.000000')
        self.__display = tkinter.Label(self, textvariable=self.__time)
        self.__button = tkinter.Button(self, text='Start', command=self.__click)
        self.__label.grid(row=0, column=0, sticky=tkinter.E, **padding)
        self.__display.grid(row=0, column=1, sticky=tkinter.EW, **padding)
        self.__button.grid(row=1, column=0, columnspan=2,
                           sticky=tkinter.NSEW, **padding)

    def __click(self):
        if self.__button['text'] == 'Start':
            self.__button['text'] = 'Stop'
            self.__start = time.clock()
            self.__counter = self.after_idle(self.__update)
        else:
            self.__button['text'] = 'Start'
            self.after_cancel(self.__counter)
```

Another example, expanding on the original code, could be a digital stopwatch. For that you could use the following:

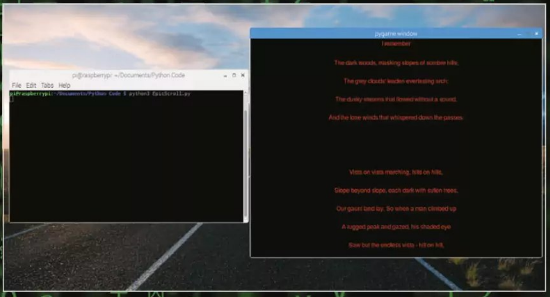
```
import tkinter
import time

class StopWatch(tkinter.Frame):

    @classmethod
    def main(cls):
        tkinter.NoDefaultRoot()
        root = tkinter.Tk()


      root.title('Stop Watch')
      root.resizable(True, False)
      root.grid _ columnconfigure(0, weight=1)
      padding = dict(padx=5, pady=5)
      widget = StopWatch(root, **padding)
      widget.grid(sticky=tkinter.NSEW, **padding)
      root.mainloop()

   def _ _init _ _(self, master=None, cnf={}, **kw):
      padding = dict(padx=kw.pop('padx', 5), pady=kw.
      pop('pady', 5))
      super(). _ _init _ _(master, cnf, **kw)
      self.grid _ columnconfigure(1, weight=1)
      self.grid _ rowconfigure(1, weight=1)
           self. _ _total = 0
      self. _ _label = tkinter.Label(self,
      text='Total Time:')
      self. _ _time = tkinter.StringVar(self,
       '0.000000')
      self. _ _display = tkinter.Label(self,
      textvariable=self. _ _time)
      self. _ _button = tkinter.Button(self,
      text='Start', command=self. _ _click)
      self. _ _label.grid(row=0, column=0,
      sticky=tkinter.E, **padding)
      self. _ _display.grid(row=0, column=1,
      sticky=tkinter.EW, **padding)
      self. _ _button.grid(row=1, column=0,
      columnspan=2,sticky=tkinter.NSEW, **padding)

   def _ _click(self):
      if self. _ _button['text'] == 'Start':
      self. _ _button['text'] = 'Stop'
      self. _ _start = time.clock()
      self. _ _counter = self.after _ idle(self. _ _update)
   else:
      self. _ _button['text'] = 'Start'
      self.after _ cancel(self. _ _counter)

   def _ _update(self):
      now = time.clock()
      diff = now - self. _ _start
      self. _ _start = now
      self. _ _total += diff
      self. _ _time.set('{:.6f}'.format(self. _ _total))
      self. _ _counter = self.after _ idle(self. _ _update)

if _ _name _ _ == '_ _main _ _':
   StopWatch.main()
```

Vertically Scrolling Text

What's not to like about vertically scrolling text? Its uses are many: the beginning of a game or introduction to something epic, like the beginning of every Star Wars movie; a list of credits at the end of something, such as a Python presentation. The list goes on.


pygame window
EpicScroll.py - /home/pi/Documents/Python Code/EpicScroll.py (3.4.2)
File Edit Format Run Options Windows Help

```
The dark woods, masking slopes of sombre hills;
The grey clouds' leaden everlasting arch;
The dusky streams that flowed without a sound,
And the lone winds that whispered down the passes.

Vista on vista marching, hills on hills,
Slope beyond slope, each dark with sullen trees,
Our gaunt land lay. So when a man climbed up
A rugged peak and gazed, his shaded eye
Saw but the endless vista - hill on hill,
Slope beyond slope, each hooded like its brothers.
```



```
pi@raspberrypi:~/Documents/Python Code $ python3 EpicScroll.py
```

```
I remember

The dark woods, masking slopes of sombre hills;
The grey clouds' leaden everlasting arch;
The dusky streams that flowed without a sound,
And the lone winds that whispered down the passes.

Vista on vista marching, hills on hills,
Slope beyond slope, each dark with sullen trees,
Our gaunt land lay. So when a man climbed up
A rugged peak and gazed, his shaded eye
Saw but the endless vista - hill on hill,
```

EPICSCROLL.PY

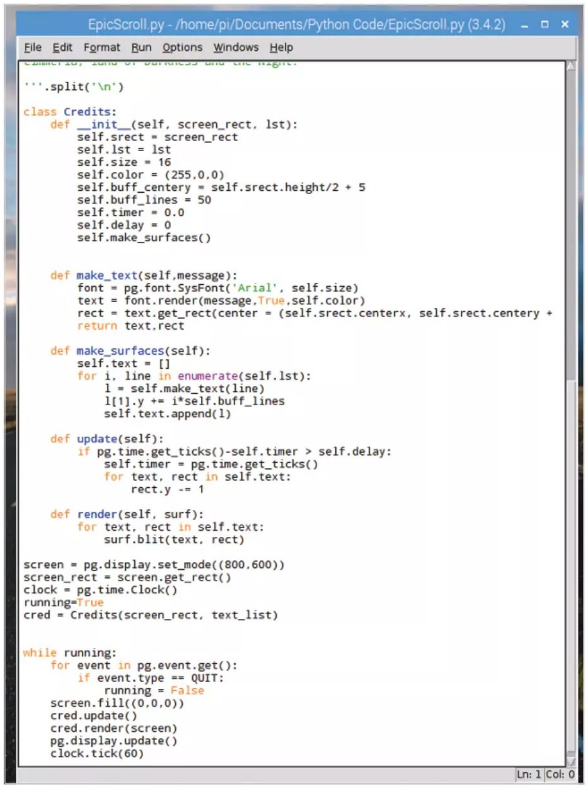
We've used the poem Cimmeria by Robert E. Howard for the code's scrolling text, along with a dramatic black background and red text. We think you'll agree, it's quite epic.

```
import pygame as pg
from pygame.locals import *

pg.init()

text _ list = '''

I remember
The dark woods, masking slopes of sombre hills;
The grey clouds' leaden everlasting arch;
The dusky streams that flowed without a sound,
And the lone winds that whispered down the passes.


Vista on vista marching, hills on hills,
Slope beyond slope, each dark with sullen trees,
Our gaunt land lay. So when a man climbed up
A rugged peak and gazed, his shaded eye
Saw but the endless vista – hill on hill,
Slope beyond slope, each hooded like its brothers.


It was a gloomy land that seemed to hold
All winds and clouds and dreams that shun the sun,
With bare boughs rattling in the lonesome winds,
And the dark woodlands brooding over all,
Not even lightened by the rare dim sun
Which made squat shadows out of men; they called it
Cimmeria, land of Darkness and deep Night.


It was so long ago and far away
I have forgot the very name men called me.
The axe and flint-tipped spear are like a dream,
And hunts and wars are shadows. I recall
Only the stillness of that sombre land;
The clouds that piled forever on the hills,
The dimness of the everlasting woods.
Cimmeria, land of Darkness and the Night.


Oh, soul of mine, born out of shadowed hills,
To clouds and winds and ghosts that shun the sun,
How many deaths shall serve to break at last
This heritage which wraps me in the grey
Apparel of ghosts? I search my heart and find
Cimmeria, land of Darkness and the Night!

 '''.split('\n')
```

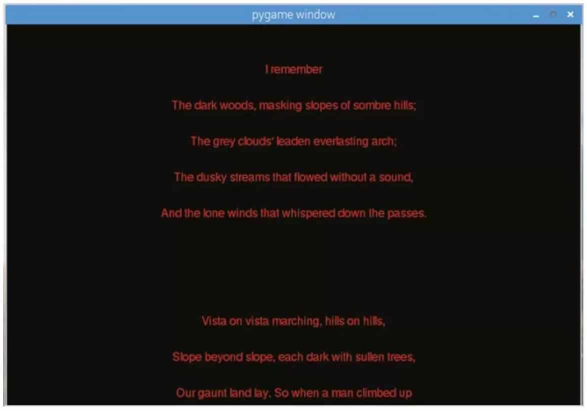
```
class Credits:
    def _ _ init _ _(self, screen _ rect, lst):
        self.srect = screen _ rect
        self.lst = lst
        self.size = 16
        self.color = (255,0,0)
        self.buff _ centery = self.srect.height/2 + 5
        self.buff _ lines = 50
        self.timer = 0.0
        self.delay = 0
        self.make _ surfaces()


    def make _ text(self,message):
        font = pg.font.SysFont('Arial', self.size)
        text = font.render(message,True,self.color)
        rect = text.get _ rect(center = (self.srect.
        centerx, self.srect.centery + self.buff _ centery) )
        return text,rect

    def make _ surfaces(self):
        self.text = []
        for i, line in enumerate(self.lst):
            l = self.make _ text(line)
            l[1].y += i*self.buff _ lines
            self.text.append(l)

    def update(self):
        if pg.time.get _ ticks()-self.timer > self.delay:
            self.timer = pg.time.get _ ticks()
            for text, rect in self.text:
                rect.y -= 1

    def render(self, surf):
        for text, rect in self.text:
            surf.blit(text, rect)

screen = pg.display.set _ mode((800,600))
screen _ rect = screen.get _ rect()
clock = pg.time.Clock()
running=True
cred = Credits(screen _ rect, text _ list)


while running:
    for event in pg.event.get():
        if event.type == QUIT:
            running = False
    screen.fill((0,0,0))
    cred.update()
    cred.render(screen)
    pg.display.update()
    clock.tick(60)
```

A Long Time Ago...

The obvious main point of enhancement is the actual text itself. Replace it with a list of credits, or an equally epic opening storyline to your Python game, and it will certainly hit the mark with whoever plays it. Don't forget to change the screen resolution if needed; we're currently running it at 800 x 600.


EpicScroll.py - /home/pi/Documents/Python Code/EpicScroll.py (3.4.2)
File Edit Format Run Options Windows Help
Ln: 1 Col: 0

```
'''.split('\n')

class Credits:
    def __init__(self, screen_rect, lst):
        self.srect = screen_rect
        self.lst = lst
        self.size = 16
        self.color = (255,0,0)
        self.buff_centery = self.srect.height/2 + 5
        self.buff_lines = 50
        self.timer = 0.0
        self.delay = 0
        self.make_surfaces()


    def make_text(self,message):
        font = pg.font.SysFont('Arial', self.size)
        text = font.render(message,True,self.color)
        rect = text.get_rect(center = (self.srect.centerx, self.srect.centery +
        return text,rect

    def make_surfaces(self):
        self.text = []
        for i, line in enumerate(self.lst):
            l = self.make_text(line)
            l[1].y += i*self.buff_lines
            self.text.append(l)

    def update(self):
        if pg.time.get_ticks()-self.timer > self.delay:
            self.timer = pg.time.get_ticks()
            for text, rect in self.text:
                rect.y -= 1

    def render(self, surf):
        for text, rect in self.text:
            surf.blit(text, rect)

screen = pg.display.set_mode((800,600))
screen_rect = screen.get_rect()
clock = pg.time.Clock()
running=True
cred = Credits(screen_rect, text_list)


while running:
    for event in pg.event.get():
        if event.type == QUIT:
            running = False
    screen.fill((0,0,0))
    cred.update()
    cred.render(screen)
    pg.display.update()
    clock.tick(60)
```



```
I remember

The dark woods, masking slopes of sombre hills;

The grey clouds' leaden everlasting arch;

The dusky streams that flowed without a sound,

And the lone winds that whispered down the passes.



Vista on vista marching, hills on hills,

Slope beyond slope, each dark with sullen trees,

Our gaunt land lay. So when a man climbed up
```

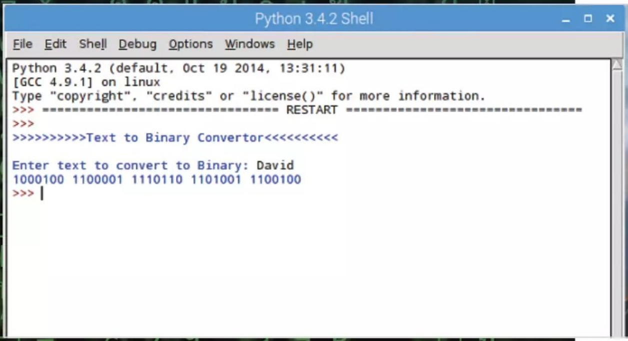
Text to Binary Convertor

While it may not seem too exciting, this text to binary convertor is actually quite good fun. It also only uses two lines of code, so it's extremely easy to insert into your own script.

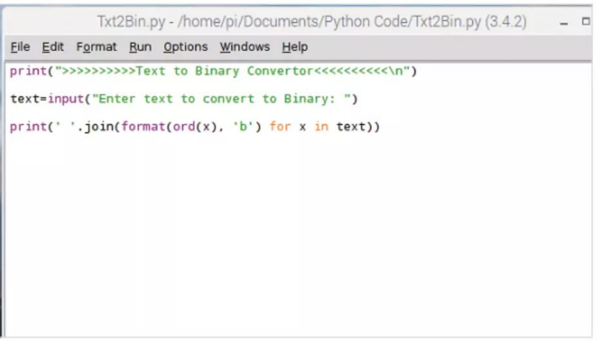
TXT2BIN.PY

Naturally we're using the format function to convert the user's entered text string into its binary equivalent. If you want to check its accuracy, you can plug the binary into an online convertor.

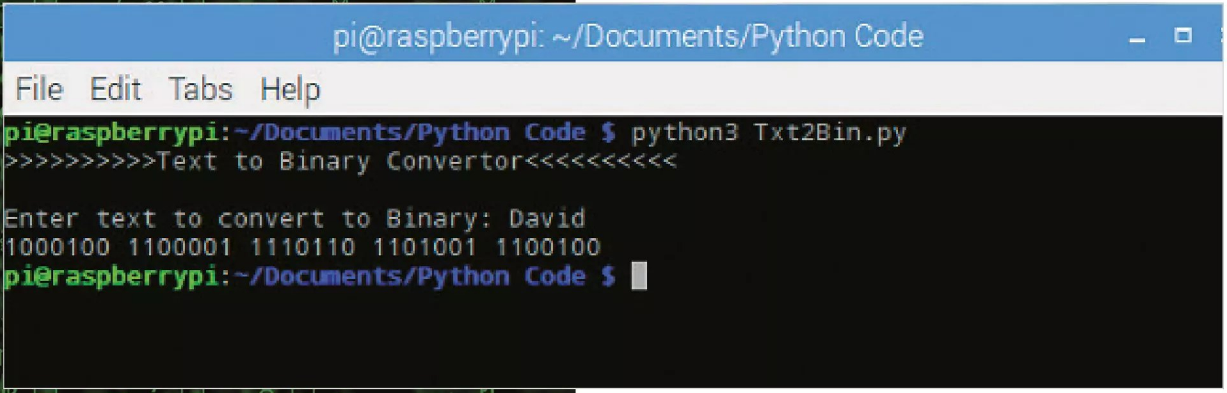
```
text=input("Enter text to convert to Binary: ")

print(' '.join(format(ord(x), 'b') for x in text))
```



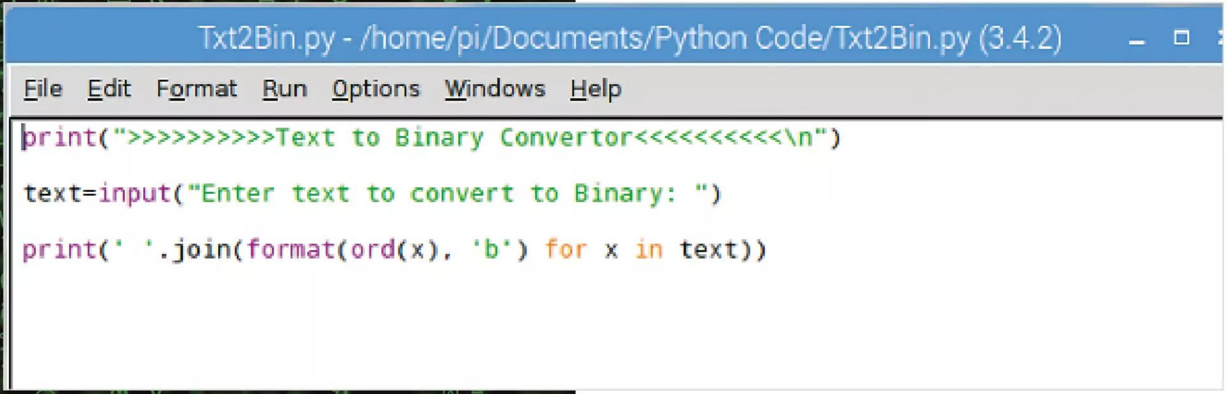
```
Python 3.4.2 (default, Oct 19 2014, 13:31:11)
[GCC 4.9.1] on linux
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
>>>>>>>>>>Text to Binary Convertor<<<<<<<<<<

Enter text to convert to Binary: David
1000100 1100001 1110110 1101001 1100100
>>>
```



```
print(">>>>>>>>>>Text to Binary Convertor<<<<<<<<<<\n")

text=input("Enter text to convert to Binary: ")

print(' '.join(format(ord(x), 'b') for x in text))
```



```
pi@raspberrypi:~/Documents/Python Code $ python3 Txt2Bin.py
>>>>>>>>>>Text to Binary Convertor<<<<<<<<<<

Enter text to convert to Binary: David
1000100 1100001 1110110 1101001 1100100
pi@raspberrypi:~/Documents/Python Code $
```



```
print(">>>>>>>>>>Text to Binary Convertor<<<<<<<<<<\n")

text=input("Enter text to convert to Binary: ")

print(' '.join(format(ord(x), 'b') for x in text))
```

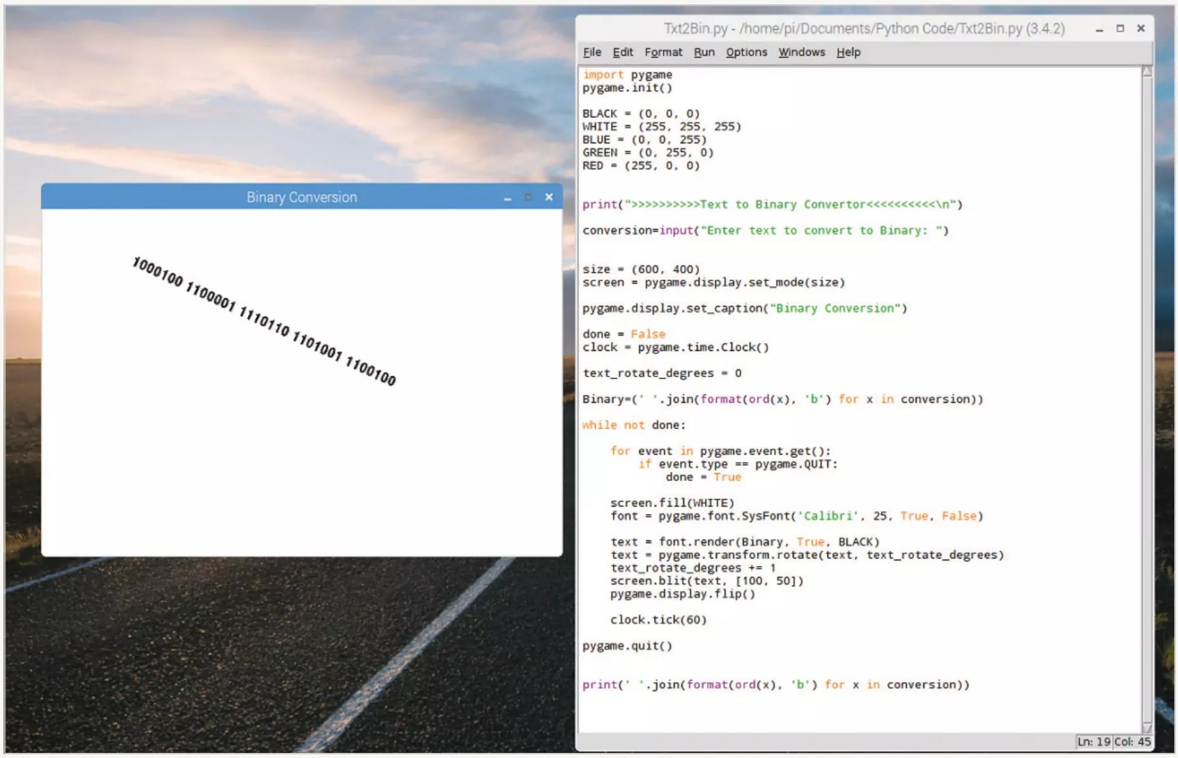
1000010 1101001 1101110 1100001 1110010 1111001

The text to binary convertor does offer some room for improvement and enhancement. There are many uses: it could be utilised in a password or secret word script, as part of an adventure game or just a novel way to display someone's name.

With regards to improvements, you could display the binary conversion in a Pygame window, using the animated text options from page 100. You could also ask the user if they wanted to have another go, or even ask if they wanted the binary output to be saved to a file.

With regards to rendering the outputted binary conversion to a Pygame window, complete with rotating text, you can use:

```
import pygame
pygame.init()

BLACK = (0, 0, 0)
WHITE = (255, 255, 255)
BLUE = (0, 0, 255)
GREEN = (0, 255, 0)
RED = (255, 0, 0)

print(“>>>>>>>>>>Text to Binary Convertor<<<<<<<<<<\n”)

conversion=input(“Enter text to convert to Binary: “)

size = (600, 400)
screen = pygame.display.set _ mode(size)

pygame.display.set _ caption(“Binary Conversion”)

done = False
clock = pygame.time.Clock()

text _ rotate _ degrees = 0

Binary=(‘ ‘.join(format(ord(x), ‘b’) for x
   in conversion))

while not done:

   for event in pygame.event.get():
       if event.type == pygame.QUIT:
           done = True

   screen.fill(WHITE)
   font = pygame.font.SysFont(‘Calibri’, 25, True, False)

   text = font.render(Binary, True, BLACK)
   text = pygame.transform.rotate(text, text _
   rotate _ degrees)
   text _ rotate _ degrees += 1
   screen.blit(text, [100, 50])
   pygame.display.flip()

   clock.tick(60)

pygame.quit()

print(‘ ‘.join(format(ord(x), ‘b’) for x in conversion))
```


Binary Conversion

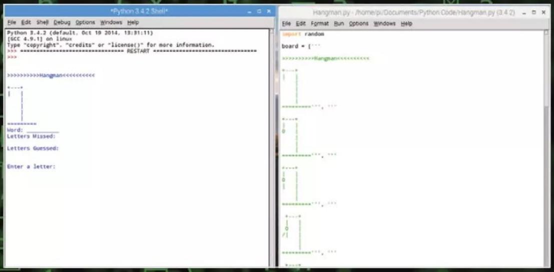
1000100 1100001 1110110 1101001 1100100

Txt2Bin.py - /home/pi/Documents/Python Code/Txt2Bin.py (3.4.2)

```
import pygame
pygame.init()

BLACK = (0, 0, 0)
WHITE = (255, 255, 255)
BLUE = (0, 0, 255)
GREEN = (0, 255, 0)
RED = (255, 0, 0)


print(">>>>>>>>>>Text to Binary Convertor<<<<<<<<<<\n")

conversion=input("Enter text to convert to Binary: ")


size = (600, 400)
screen = pygame.display.set_mode(size)

pygame.display.set_caption("Binary Conversion")

done = False
clock = pygame.time.Clock()

text_rotate_degrees = 0

Binary=(' '.join(format(ord(x), 'b') for x in conversion))

while not done:

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            done = True

    screen.fill(WHITE)
    font = pygame.font.SysFont('Calibri', 25, True, False)

    text = font.render(Binary, True, BLACK)
    text = pygame.transform.rotate(text, text_rotate_degrees)
    text_rotate_degrees += 1
    screen.blit(text, [100, 50])
    pygame.display.flip()

    clock.tick(60)

pygame.quit()


print(' '.join(format(ord(x), 'b') for x in conversion))
```

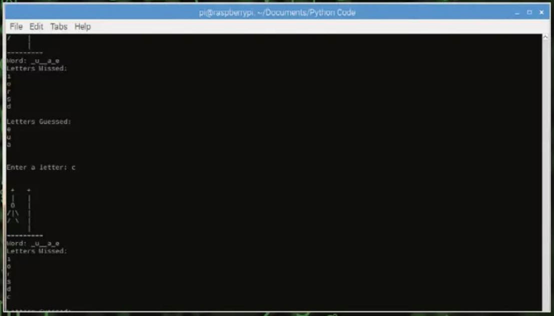
Hangman Game Script

Hangman is a great game to program into Python. It can be extremely complex, displaying graphics, the number of guesses left in the secret word, a huge bank of available words picked at random and countless other elements. It can also be quite simple. Here we have a mix between the two.

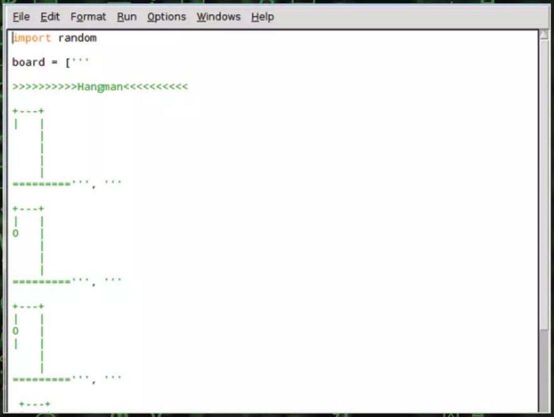


```
Python 3.4.2 (default, Oct 19 2014, 13:31:11)
[GCC 4.9.1] on linux
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>


>>>>>>>>>>Hangman<<<<<<<<<<

+---+
|   |
    |
    |
    |
    |
=========
Word: __________
Letters Missed:

Letters Guessed:


Enter a letter:
```

```
import random

board = ['''

>>>>>>>>>>Hangman<<<<<<<<<<
```



```
/   |
    |
---------
Word: _u__a_e
Letters Missed:
i
o
r
s
d

Letters Guessed:
e
u
a


Enter a letter: c


 +   +
 |   |
 O   |
/|\  |
/ \  |
     |
---------
Word: _u__a_e
Letters Missed:
i
o
r
s
d
c
```



```
import random

board = ['''

>>>>>>>>>>Hangman<<<<<<<<<<

+---+
|   |
    |
    |
    |
    |
=========''', '''

+---+
|   |
O   |
    |
    |
    |
=========''', '''

+---+
|   |
O   |
|   |
    |
    |
=========''', '''

 +---+
```

HANGMAN.PY

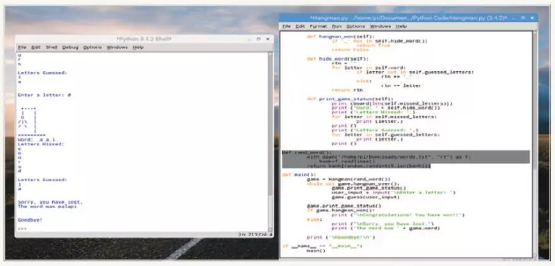
We've made a Hangman game board (the gallows) out of characters that can be displayed in the IDLE Shell, along with a huge bank of words to randomly choose from.

```
import random

board = ['''

>>>>>>>>>>Hangman<<<<<<<<<<

+---+
|   |
    |
    |
    |
    |
=========''', '''

+----+
|    |
O   |
     |
     |
     |
=========''', '''

+---+
|    |
O   |
|    |
     |
     |
=========''', '''

 +---+
 |   |
 O   |
/|   |
    |
    |
=========''', '''

 +---+
 |     |
 O    |
/|\  |
       |
       |
=========''', '''

 +---+
 |     |
 O    |
/|\  |
/      |
         |
=========’’’, ‘’’

 +---+
 |   |
 O   |
/|\  |
/ \  |
         |
=========’’’]


class Hangman:
   def __init__(self,word):
      self.word = word
      self.missed_letters = []
      self.guessed_letters = []

   def guess(self,letter):
      if letter in self.word and letter not in self.
      guessed_letters:
         self.guessed_letters.append(letter)
      elif letter not in self.word and letter not in
      self.missed_letters:
         self.missed_letters.append(letter)
      else:
         return False
      return True

   def hangman_over(self):
      return self.hangman_won() or (len(self.missed_
      letters) == 6)

   def hangman_won(self):
      if ‘_’ not in self.hide_word():
         return True
      return False

   def hide_word(self):
      rtn = ‘’
      for letter in self.word:
         if letter not in self.guessed_letters:
            rtn += ‘_’
         else:
            rtn += letter
      return rtn

   def print_game_status(self):
      print (board[len(self.missed_letters)])
      print (‘Word: ‘ + self.hide_word())
      print (‘Letters Missed: ‘,)
      for letter in self.missed_letters:
         print (letter,)
      print ()
      print (‘Letters Guessed: ‘,)
      for letter in self.guessed_letters:
         print (letter,)
      print ()

def rand_word():
   bank = ‘ability about above absolute accessible
   accommodation accounting beautiful bookstore
   calculator clever engaged engineer enough
   handsome refrigerator opposite socks interested
   strawberry backgammon anniversary confused
   dangerous entertainment exhausted impossible
   overweight temperature vacation scissors
   accommodation appointment decrease development
   earthquake environment brand environment necessary
   luggage responsible ambassador circumstance
   congratulate frequent’.split()
   return bank[random.randint(0,len(bank))]

def main():
   game = Hangman(rand_word())
   while not game.hangman_over():
      game.print_game_status()
      user_input = input(‘\nEnter a letter: ‘)
      game.guess(user_input)

   game.print_game_status()
   if game.hangman_won():
      print (‘\nCongratulations! You have won!!’)
   else:
      print (‘\nSorry, you have lost.’)
      print (‘The word was ‘ + game.word)

   print (‘\nGoodbye!\n’)

if __name__ == “__main__”:
   main()
```

QUIT()

Since this is the last example in our Python code repository, we thought we’d go out with a bang and feature the hangman gallows being drawn with each incorrect guess of the word. Don’t worry if it looks misaligned in the text here, this is merely due to the differences between using the Python IDLE editor and pasting the code into a word processor (which formats things differently).

There’s plenty you can do to improve, enhance and expand on what we’ve presented here. You can include a routine that returns an error if the user enters a number or character. You can include extra points for someone who guesses the entire word in one go rather than one letter at a time and you could perhaps add Chopin’s Funeral March should you lose the game; or something celebratory if you win.



Consider replacing the bank of words too. They’re found under the bank list, and could easily be swapped out for something more difficult. If you download www.github.com/dwyl/english-words you can find a text document with over 466,000 words. Perhaps you could swap the words in the bank to instead read the contents of the text file:

```
def rand_word():
   with open(“/home/pi/Downloads/words.txt”, “rt”) as f:
      bank=f.readlines()
   return bank[random.randint(0,len(bank))]
```

Mastering Linux

The Raspberry Pi OS is Linux based, which is an incredibly powerful operating system that drives many supercomputers, web servers and even top-end military hardware. Get to grips with Linux, and you will not just master the Pi but also the power behind the Internet.

In this section, you will discover how the OS works, how the filesystem is built and how you can list, move, create and delete files and folders. To truly be able to hack the Raspberry Pi, you will need to be familiar with the OS and its inner-workings.

The more adept you become at Linux, the better your projects will be, and the more power you'll have over them and how they interact with the digital world to which they're connected.

What is Linux?

The Raspberry Pi operating system is Raspbian, which is a Linux operating system; but what exactly is Linux? Where did it come from and what does it do? In a world where Windows and macOS have supremacy of the desktop, it's easy to overlook it, but there's more to Linux than you might imagine.

Linux is a surprisingly powerful, fast, secure and capable operating system. It's used as the OS of choice for the Raspberry Pi, in the form of Raspbian OS, as well as in some of the most unlikely places.

Despite only enjoying a 1.96% share (according to netmarketshare.com) of the total desktop operating system market, Linux has a dedicated following of enthusiasts, users and contributors. It was created in 1991 by University of Helsinki student, Linus Torvalds, who had become frustrated with the limitations and licensing of the popular educational system Minix, a miniature version of the Unix operating system, in use at the time.

Unix itself was released in the early '70s, as a multi-tasking, modular-designed operating system originally developed for programmers who needed a stable platform to code on. However, its performance, power and portability meant that it soon became the system of choice for companies and universities where high-end computing tasks were needed.

Torvalds needed a system that could mirror Unix's performance and features, without the licensing cost. Thus was born Linux, the Unix-like operating system which used freely available code from the GNU project. This enabled users around the world to utilise the power of the Unix-like system, completely free of charge, an ethos that still holds today: Linux is free to download, install and use.

Linux is much like any other operating system, such as Windows or macOS in that it manages the computer hardware, provides an interface for the user to access that hardware and comes with programs for productivity, communications, gaming, science, education and more. Linux can be broken up into a number of significant elements:

BOOTLOADER

The bootloader is the software that initialises and boots up your computer. It loads up the various modules the OS uses to begin to access the hardware in the system.

KERNEL

The kernel is the core of the system and the single element that is actually called Linux. The Linux kernel manages the computer processor, memory, storage and any peripherals you have attached to your computer.

DAEMONS

Daemons are background services that start as the operating system is booting. These can enable printing, sound, networking and so on.

GRAPHICAL SERVER

This is a module within Linux that provides a graphical output to your monitor. It's referred to as the X server or simply X.

SHELL

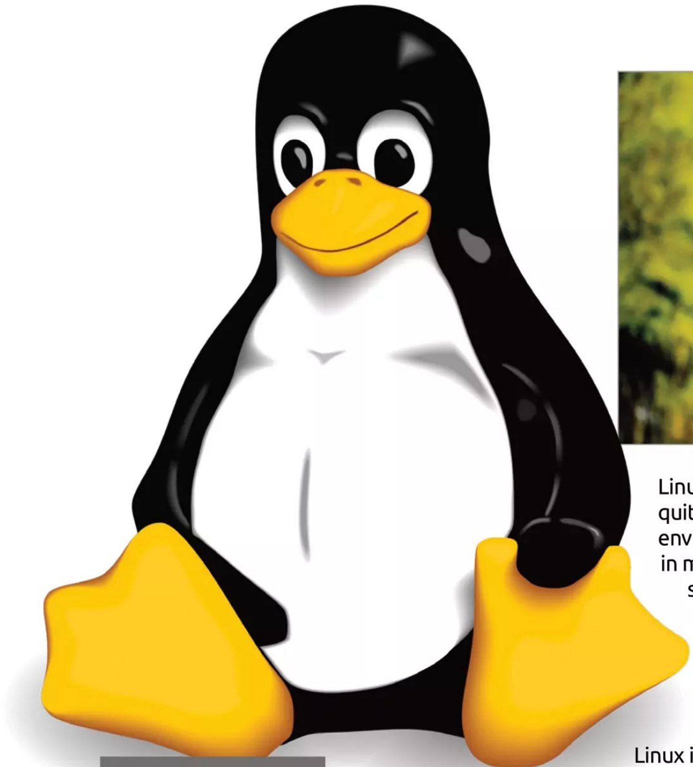
The Linux shell is a command line interface environment, which a Linux user can use to enter commands to the OS that directly affect it. Within the shell you can add new users, reboot the system, create and delete files and folders plus much more. BASH (Bourne-Again Shell) is the most popular shell used in Linux, although others are available. The shell is also known as the Terminal and it's where we're going to work from through this section of the book.

DESKTOP ENVIRONMENT

The Desktop Environment, or DE, is the main Graphical User Interface (GUI) that users interact with. It's the desktop and includes Internet browsers, productivity, games and whatever program or app you're using. There are countless DEs available, however Raspbian uses PIXEL.

PROGRAMS/APPLICATIONS

As Linux is a free, open source operating system, it also makes good use of the tens of thousands of freely available applications. The likes of LibreOffice, GIMP and Python are just the tip of the iceberg.



Tux, the Linux mascot (Linus likes penguins).

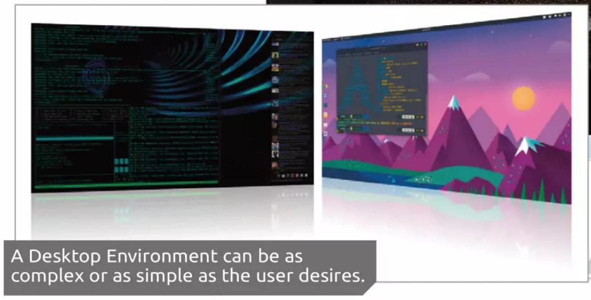
Linus Torvalds, the creator of the Linux kernel.

Linux is used throughout the world, in a number of basic and quite unique uses. While it may look radically different from one environment to the next, the actual Linux kernel, can be found in modern smart TVs, in-car entertainment systems and GPS, supercomputers, IoT devices and the Raspberry Pi. It's used by NASA, both in the command centre and on-board the ISS. Linux servers power the backbone of the Internet, along with most of the websites you visit daily. Android utilises components of the Linux kernel, as do set top boxes, games consoles and even your fridge, freezer, oven and washing machine.

Linux isn't just a free to use operating system. It's stable, powerful and fast, easily customised and requires very little maintenance. However, it's more than just performance stats; Linux means freedom from the walled garden approach of other operating systems. It's a lively community of like-minded individuals who want more from their computers without the shackles of price or conformity. Linux means choice.

```
Ign http://archive.raspberrypi.org jessie/main Translation-en_GB
Get:14 https://download.sublimetext.com apt/dev/ Translation-en_GB [1,130 B]
Ign http://archive.raspberrypi.org jessie/main Translation-en
Get:15 https://download.sublimetext.com apt/dev/ Translation-en [1,130 B]
Ign http://archive.raspberrypi.org jessie/ui Translation-en_GB
Ign http://archive.raspberrypi.org jessie/ui Translation-en
Get:16 https://download.sublimetext.com apt/dev/ Translation-en_GB [1,130 B]
Ign https://download.sublimetext.com apt/dev/ Translation-en_GB
Get:17 https://download.sublimetext.com apt/dev/ Translation-en [1,130 B]
Ign https://download.sublimetext.com apt/dev/ Translation-en
Get:18 http://mirrordirector.raspbian.org jessie/contrib armhf Packages [43.3 kB
]
Get:19 http://mirrordirector.raspbian.org jessie/non-free armhf Packages [84.2 k
B]
Get:20 http://mirrordirector.raspbian.org jessie/rpi armhf Packages [1,356 B]
Ign http://mirrordirector.raspbian.org jessie/contrib Translation-en_GB
Ign http://mirrordirector.raspbian.org jessie/contrib Translation-en
Ign http://mirrordirector.raspbian.org jessie/main Translation-en_GB
Ign http://mirrordirector.raspbian.org jessie/main Translation-en
Ign http://mirrordirector.raspbian.org jessie/non-free Translation-en_GB
Ign http://mirrordirector.raspbian.org jessie/non-free Translation-en
Ign http://mirrordirector.raspbian.org jessie/rpi Translation-en_GB
Ign http://mirrordirector.raspbian.org jessie/rpi Translation-en
100% [3 Packages xz 0 B]                                        1,542 kB/s 0s
```

Raspbian on the Raspberry Pi, is the Linux distribution of choice.



A Desktop Environment can be as complex or as simple as the user desires.

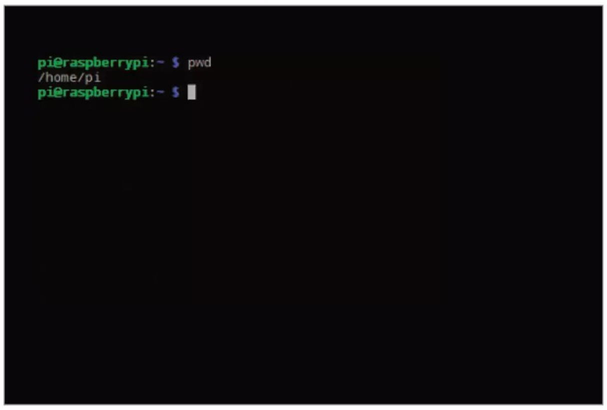
Using the Filesystem

To master Linux, it's important to understand how the filesystem works. What's more, it's also important to become familiar with the Terminal, or shell. This command line environment may appear daunting at first, but with practise, it soon becomes easy to use.

GETTING AROUND

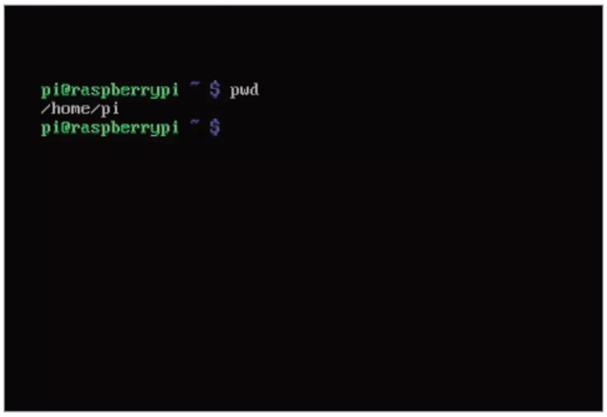
To drop into the Terminal, click on the fourth icon from the left along the top of the Raspberry Pi desktop, the one with a right-facing arrow and an underscore. This is the shell, or Terminal.

STEP 1 First, you're going to look at directories and the directory path. A directory is the same thing as a folder, however in Linux it's always called a directory. These are placed inside each other using a "/" character. So when you see /home/pi it means the pi directory is inside the home directory. Enter: `clear` and press return to clean the screen. Now enter: `pwd`. This stands for Print Working Directory and displays /home/pi.



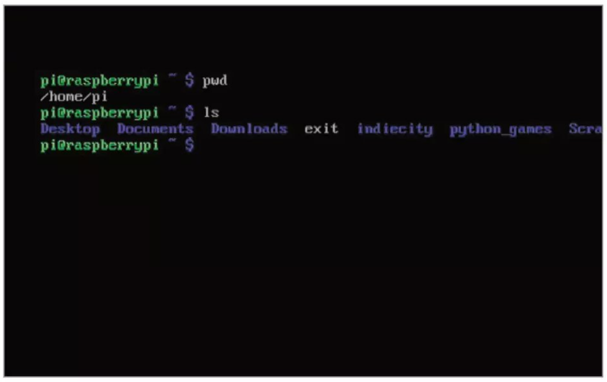
```
pi@raspberrypi:~ $ pwd
/home/pi
pi@raspberrypi:~ $ ▌
```

STEP 2 When you log in to your Raspberry Pi, you don't start at the base of the hard drive, known as the 'root' (also known as the topmost directory). Instead you begin inside your user directory, which is named 'pi' by default and is itself in a directory called 'home'. Directories are indicated by the '/' symbol. So, "'/home/pi'" tells you that in the root is a directory called home, and the next "'/'" says that inside "home" is a directory called "pi". That's where you start.



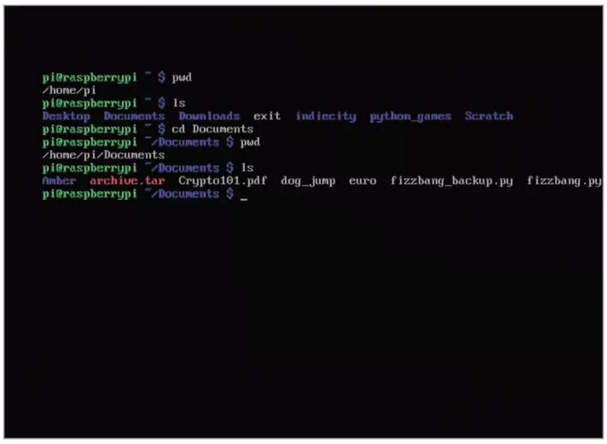
```
pi@raspberrypi ~ $ pwd
/home/pi
pi@raspberrypi ~ $
```

STEP 3 Enter: `ls` to view the contents of the current directory. You should see Desktop, Documents, and Downloads and Scratch in Blue. You may also see other items depending on how much you have used your Raspberry Pi. The colour code is worth knowing: directories are blue while most files are white. As you go on you'll see other colours: executable files (programs) are bright green, archived files are red and so on. Blue and white are the two you need to know to get started.



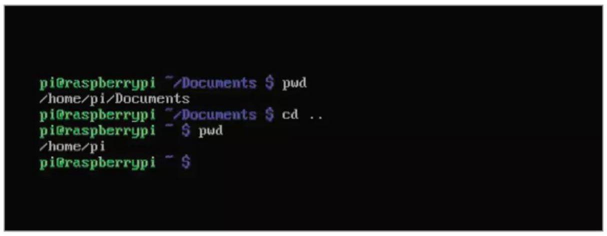
```
pi@raspberrypi ~ $ pwd
/home/pi
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  exit  indiecity  python_games  Scra
pi@raspberrypi ~ $
```

STEP 4 Now you're going to move from the pi directory into the Documents directory. Enter: `cd Documents`. Note the capital "D". Linux is case sensitive, which means you have to enter the exact name including correct capitalisation. The cd command stands for change directory. Now enter: `pwd` again to view the directory path. It will display /home/pi/ Documents. Enter: `ls` to view the files inside the Documents directory.

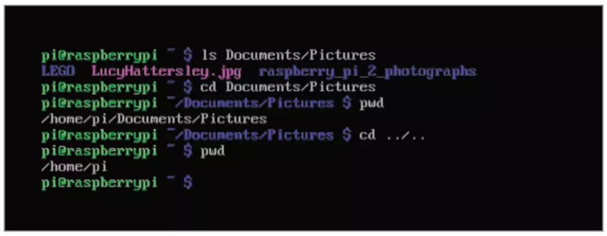


```
pi@raspberrypi ~ $ pwd
/home/pi
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  exit  indiecity  python_games  Scratch
pi@raspberrypi ~ $ cd Documents
pi@raspberrypi ~/Documents $ pwd
/home/pi/Documents
pi@raspberrypi ~/Documents $ ls
Amber  archive.tar  Crypto101.pdf  dog_jump  euro  fizzbang_backup.py  fizzbang.py
pi@raspberrypi ~/Documents $ _
```

STEP 5 How do you get back up to the pi directory? By using a command "cd ..". In Linux two dots means the directory above, also known as the parent directory. Incidentally, a single dot "." is used for the same directory. You never use "cd ." to switch to the same directory but it's worth knowing because some commands need you to specify the current directory.


```
pi@raspberrypi ~/Documents $ pwd
/home/pi/Documents
pi@raspberrypi ~/Documents $ cd ..
pi@raspberrypi ~ $ pwd
/home/pi
pi@raspberrypi ~ $
```

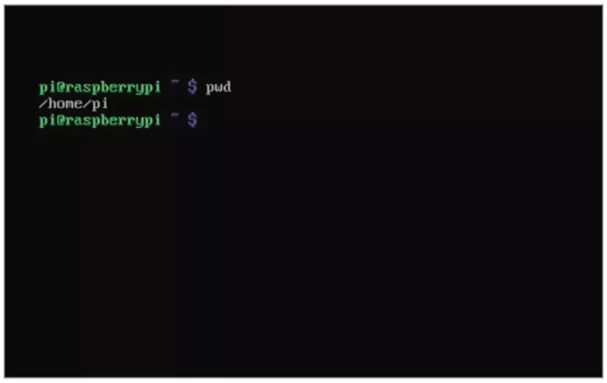
STEP 6 The "ls" and "cd" commands can also be used with more complex paths. Enter: `ls Documents/Pictures` to view the contents of a Pictures directory inside your Documents directory. You can switch to this directory using `cd Documents/Pictures`; use `cd ../..` to move back up two parent directories.


```
pi@raspberrypi ~ $ ls Documents/Pictures
LEGO  LucyHattersley.jpg  raspberry_pi_2_photographs
pi@raspberrypi ~ $ cd Documents/Pictures
pi@raspberrypi ~/Documents/Pictures $ pwd
/home/pi/Documents/Pictures
pi@raspberrypi ~/Documents/Pictures $ cd ../..
pi@raspberrypi ~ $ pwd
/home/pi
pi@raspberrypi ~ $
```

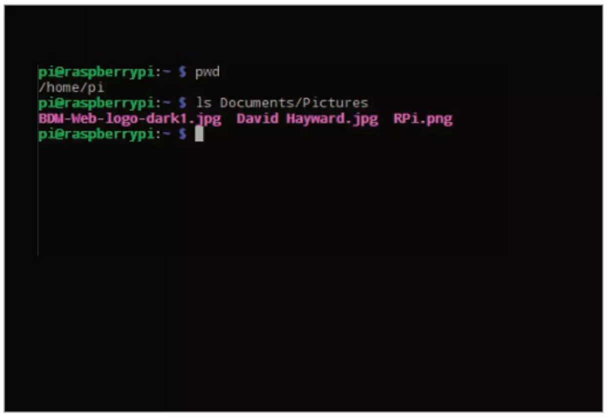
ABSOLUTE VS RELATIVE PATHS

It is important to know the difference between the working directory, root directory and home. There are also two types of path: Absolute and Relative. These are easier to understand than they sound. Let's take a look...

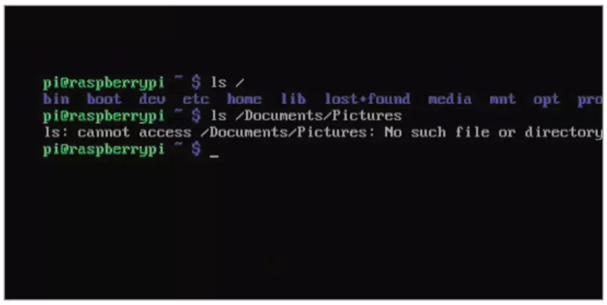
STEP 1 By default, commands like "ls" use the working directory. This is the current directory that you're looking at and is set to your home directory by default (/users/pi). Using "pwd" (Print Working Directory) lets you know what the working directory is, and using "cd" changes the working directory.


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

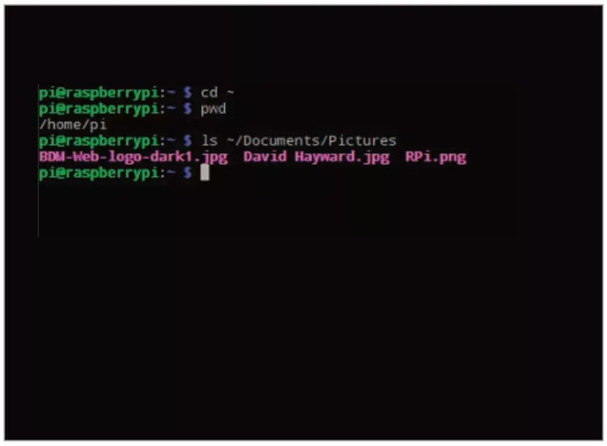
STEP 2 The root directory is always '/'. Entering: `ls /` lists the contents of root, and entering: `cd /` switches to the root directory. This is important because there is a difference between "ls Documents/Pictures" and "ls /Documents/Pictures". The first command lists the contents of the Pictures directory in Documents inside the working directory (which, if you are in the home directory, will work).


```
pi@raspberrypi:~ $ pwd
/home/pi
pi@raspberrypi:~ $ ls Documents/Pictures
BDM-Web-logo-dark1.jpg  David Hayward.jpg  RPi.png
pi@raspberrypi:~ $ █
```

STEP 3 The second command ("ls /Documents/Pictures") attempts to list the content of Pictures in a directory called Documents inside the root directory (because the path started with '/', which is root). There is typically no Documents directory in root, so you will get a "No such file or directory" error. Starting a path with '/' is known as an "absolute path", while starting without the '/' is known as a "relative path" because it is relative to your working directory.


```
pi@raspberrypi ~ $ ls /
bin  boot  dev  etc  home  lib  lost+found  media  mnt  opt  pro
pi@raspberrypi ~ $ ls /Documents/Pictures
ls: cannot access /Documents/Pictures: No such file or directory
pi@raspberrypi ~ $ _
```

STEP 4 There is also an absolute path shortcut to your user directory, and that is the tilde "~" character. Entering: `ls ~` always lists the contents of your home directory, while "cd ~" moves straight to your home directory, no matter what your working directory is. You can also use this shortcut wherever you are: enter: `ls ~/Documents/Pictures` to display the contents of the Pictures.


```
pi@raspberrypi:~ $ cd ~
pi@raspberrypi:~ $ pwd
/home/pi
pi@raspberrypi:~ $ ls ~/Documents/Pictures
BDM-Web-logo-dark1.jpg  David Hayward.jpg  RPi.png
pi@raspberrypi:~ $ █
```

Listing and Moving Files

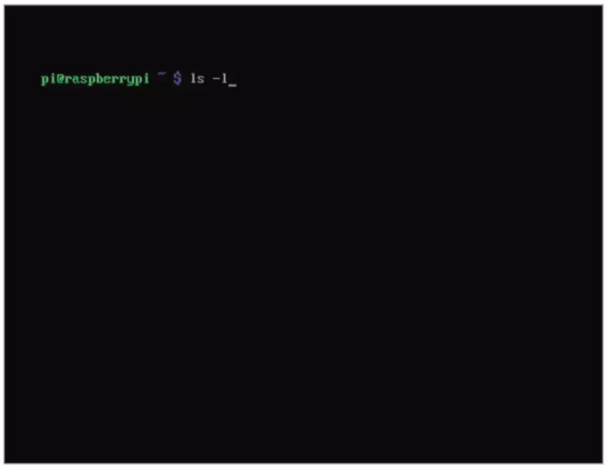
Admittedly, using the desktop GUI to list and move files is much easier than using the Terminal and keyboard. However, it's an important skill that you will appreciate as you advance with the Raspberry Pi and Linux.

LOOKING AT FILES

Operating systems are built on files and folders, or directories if you prefer. While you're used to viewing your own files, most operating systems keep other files out of sight. In Raspbian, you have access to every file in the system.

STEP 1

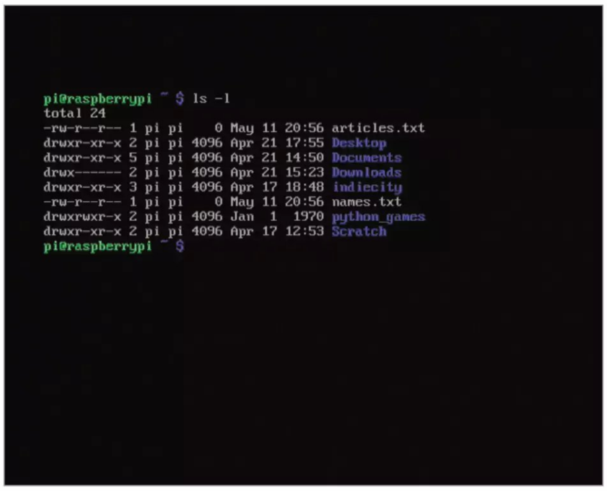
We've already looked at "ls", which lists the files in the working directory, but you are more likely to use a command like "ls –l". The bit after the command (the '-lah') is known as the argument. This is an option that modifies the behaviour of the command.



```
pi@raspberrypi ~ $ ls -l_
```

STEP 2

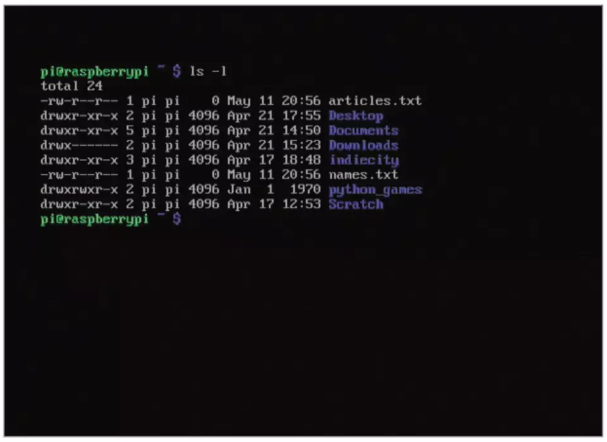
The "-l" argument lists files and directories in long format. Each file and directory is now on a single line, and before each file is a lot of text. First you'll see lots of letters and dashes, like 'drwxr-xr-x'. Don't worry about these for now; they are known as 'permissions' and we'll come to those later.



```
pi@raspberrypi ~ $ ls -l
total 24
-rw-r--r-- 1 pi pi    0 May 11 20:56 articles.txt
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 5 pi pi 4096 Apr 21 14:50 Documents
drwx------ 2 pi pi 4096 Apr 21 15:23 Downloads
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
-rw-r--r-- 1 pi pi    0 May 11 20:56 names.txt
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 Apr 17 12:53 Scratch
pi@raspberrypi ~ $
```

STEP 3

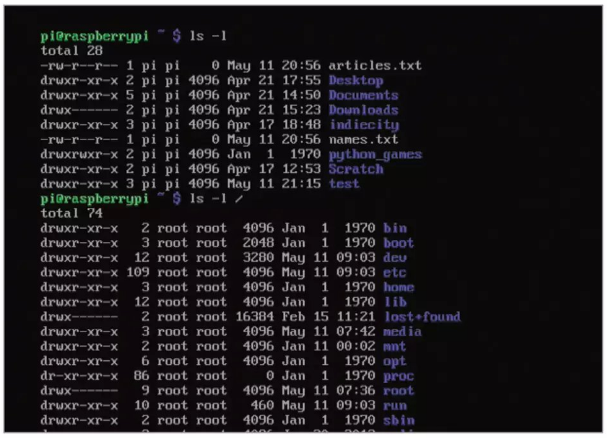
After the permission letters come a single number. This is the number of files in the item. If it's a file then it'll be 1, but if it's a directory it'll be at least 2. This is because each directory contains two hidden files; one with a single dot (.) and one with two dots (..). Directories containing files or other directories will have a higher number.



```
pi@raspberrypi ~ $ ls -l
total 24
-rw-r--r-- 1 pi pi    0 May 11 20:56 articles.txt
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 5 pi pi 4096 Apr 21 14:50 Documents
drwx------ 2 pi pi 4096 Apr 21 15:23 Downloads
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
-rw-r--r-- 1 pi pi    0 May 11 20:56 names.txt
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 Apr 17 12:53 Scratch
pi@raspberrypi ~ $
```

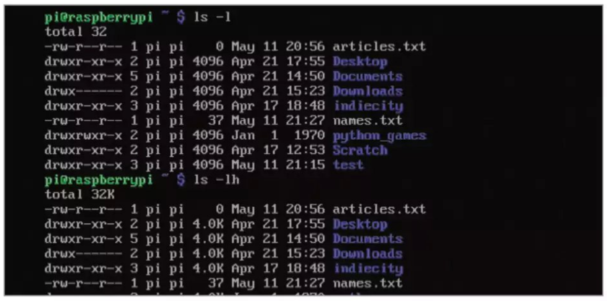
STEP 4

Next you'll see the word "pi" listed twice on each line. This refers to the user rather than the name of your computer (your default username is "pi"). The first is the owner of the file, and the second is the group. Typically these will both be the same and you'll see either 'pi' or 'root'. You can enter: `ls -l /` to view the files and directories in the root directory that belong to the root account.



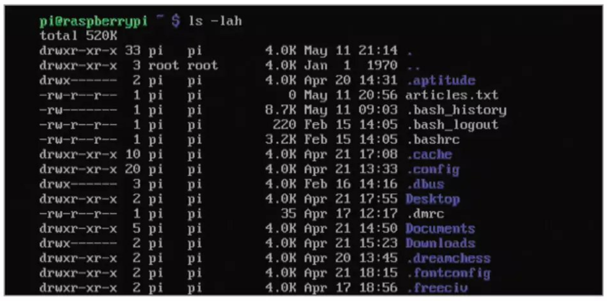
```
pi@raspberrypi ~ $ ls -l
total 28
-rw-r--r-- 1 pi pi    0 May 11 20:56 articles.txt
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 5 pi pi 4096 Apr 21 14:50 Documents
drwx------ 2 pi pi 4096 Apr 21 15:23 Downloads
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
-rw-r--r-- 1 pi pi    0 May 11 20:56 names.txt
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 Apr 17 12:53 Scratch
drwxr-xr-x 3 pi pi 4096 May 11 21:15 test
pi@raspberrypi ~ $ ls -l /
total 74
drwxr-xr-x   2 root root  4096 Jan  1  1970 bin
drwxr-xr-x   3 root root  2048 Jan  1  1970 boot
drwxr-xr-x  12 root root  3280 May 11 09:03 dev
drwxr-xr-x 109 root root  4096 May 11 09:03 etc
drwxr-xr-x   3 root root  4096 Jan  1  1970 home
drwxr-xr-x  12 root root  4096 Jan  1  1970 lib
drwx------   2 root root 16384 Feb 15 11:21 lost+found
drwxr-xr-x   3 root root  4096 May 11 07:42 media
drwxr-xr-x   2 root root  4096 Jan 11 00:02 mnt
drwxr-xr-x   6 root root  4096 Jan  1  1970 opt
dr-xr-xr-x  86 root root     0 Jan  1  1970 proc
drwx------   9 root root  4096 May 11 07:36 root
drwxr-xr-x  10 root root   460 May 11 09:03 run
drwxr-xr-x   2 root root  4096 Jan  1  1970 sbin
```

STEP 5 The next number relates to the size of the file, in bytes. In Linux each text file is made up of letters and each letter takes up a byte, so our names.txt file has 37 bytes and 37 characters in the document. Files and directories can be extremely large and hard to determine, so use "ls – lh". The "h" argument humanises the number, making it easier to read.



```
pi@raspberrypi ~ $ ls -l
total 32
-rw-r--r-- 1 pi pi    0 May 11 20:56 articles.txt
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 5 pi pi 4096 Apr 21 14:50 Documents
drwx------ 2 pi pi 4096 Apr 21 15:23 Downloads
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
-rw-r--r-- 1 pi pi   37 May 11 21:27 names.txt
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 Apr 17 12:53 Scratch
drwxr-xr-x 3 pi pi 4096 May 11 21:15 test
pi@raspberrypi ~ $ ls -lh
total 32K
-rw-r--r-- 1 pi pi    0 May 11 20:56 articles.txt
drwxr-xr-x 2 pi pi 4.0K Apr 21 17:55 Desktop
drwxr-xr-x 5 pi pi 4.0K Apr 21 14:50 Documents
drwx------ 2 pi pi 4.0K Apr 21 15:23 Downloads
drwxr-xr-x 3 pi pi 4.0K Apr 17 18:48 indiecity
-rw-r--r-- 1 pi pi   37 May 11 21:27 names.txt
```

STEP 6 Finally, you should be aware that there are many hidden files in Linux. These are listed using the "-a" argument. Hidden files and directories begin with a dot (.), so you should never start a file or directory with a dot, unless you want to hide it. Typically, you can combine all three arguments together into the command "'s –lah".

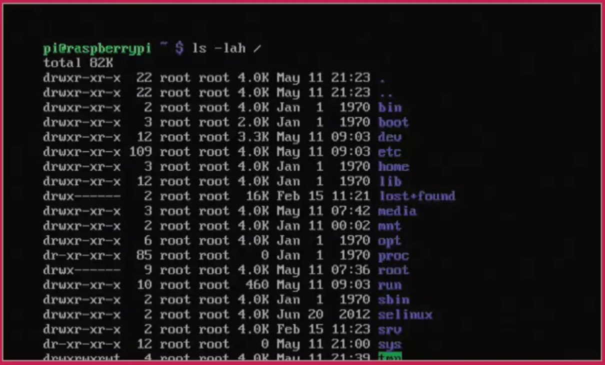


```
pi@raspberrypi ~ $ ls -lah
total 520K
drwxr-xr-x 33 pi   pi       4.0K May 11 21:14 .
drwxr-xr-x  3 root root     4.0K Jan  1  1970 ..
drwx------  2 pi   pi       4.0K Apr 20 14:31 .aptitude
-rw-r--r--  1 pi   pi          0 May 11 20:56 articles.txt
-rw-------  1 pi   pi       8.7K May 11 09:03 .bash_history
-rw-r--r--  1 pi   pi        220 Feb 15 14:05 .bash_logout
-rw-r--r--  1 pi   pi       3.2K Feb 15 14:05 .bashrc
drwxr-xr-x 10 pi   pi       4.0K Apr 21 17:08 .cache
drwxr-xr-x 20 pi   pi       4.0K Apr 21 13:33 .config
drwx------  3 pi   pi       4.0K Feb 16 14:16 .dbus
drwxr-xr-x  2 pi   pi       4.0K Apr 21 17:55 Desktop
-rw-r--r--  1 pi   pi         35 Apr 17 12:17 .dmrc
drwxr-xr-x  5 pi   pi       4.0K Apr 21 14:50 Documents
drwx------  2 pi   pi       4.0K Apr 21 15:23 Downloads
drwxr-xr-x  2 pi   pi       4.0K Apr 20 13:45 .dreamchess
drwxr-xr-x  2 pi   pi       4.0K Apr 21 18:15 .fontconfig
drwxr-xr-x  2 pi   pi       4.0K Apr 17 18:56 .freeciv
```

SOME COMMON DIRECTORIES

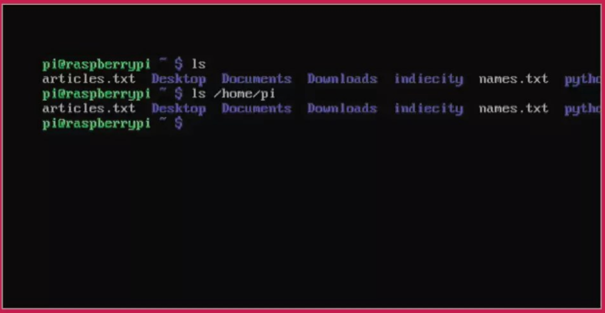
Now that you know how to view the contents of your hard drive you'll start to notice a lot of directories with names like bin, sbin, var and dev. These are the files and directories that you are kept away from on a Mac, and won't encounter on a Windows PC.

STEP 1 Enter: `ls -lah /` to view all of the files and directories, including the hidden items, in the root directory of your hard drive. Here you will see all the items that make up your Raspbian OS (which is a version of Linux). It's worth taking the time to know some of them.



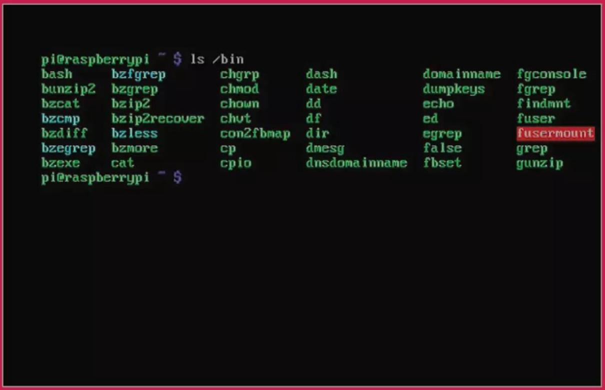
```
pi@raspberrypi ~ $ ls -lah /
total 82K
drwxr-xr-x  22 root root 4.0K May 11 21:23 .
drwxr-xr-x  22 root root 4.0K May 11 21:23 ..
drwxr-xr-x   2 root root 4.0K Jan  1  1970 bin
drwxr-xr-x   3 root root 2.0K Jan  1  1970 boot
drwxr-xr-x  12 root root 3.3K May 11 09:03 dev
drwxr-xr-x 109 root root 4.0K May 11 09:03 etc
drwxr-xr-x   3 root root 4.0K Jan  1  1970 home
drwxr-xr-x  12 root root 4.0K Jan  1  1970 lib
drwx------   2 root root  16K Feb 15 11:21 lost+found
drwxr-xr-x   3 root root 4.0K May 11 07:42 media
drwxr-xr-x   2 root root 4.0K Jan 11 00:02 mnt
drwxr-xr-x   6 root root 4.0K Jan  1  1970 opt
dr-xr-xr-x  85 root root    0 Jan  1  1970 proc
drwx------   9 root root 4.0K May 11 07:36 root
drwxr-xr-x  10 root root  460 May 11 09:03 run
drwxr-xr-x   2 root root 4.0K Jan  1  1970 sbin
drwxr-xr-x   2 root root 4.0K Jun 20  2012 selinux
drwxr-xr-x   2 root root 4.0K Feb 15 11:23 srv
dr-xr-xr-x  12 root root    0 May 11 21:00 sys
```

STEP 3 Entering: `ls /home` displays the contents of your home directory, which contains pi; the directory that you start in. So, entering: `ls/home/pi` is the same as just "ls" from the default home directory. This is where you are expected to place most of the documents you create. Don't confuse home with "usr"; the /usr directory is where find you find program tools and libraries.



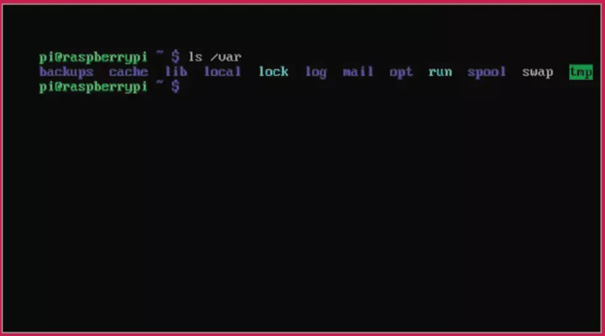
```
pi@raspberrypi ~ $ ls
articles.txt  Desktop  Documents  Downloads  indiecity  names.txt  pytho
pi@raspberrypi ~ $ ls /home/pi
articles.txt  Desktop  Documents  Downloads  indiecity  names.txt  pytho
pi@raspberrypi ~ $
```

STEP 2 Bin is a directory that stores binaries. This is the Linux way of saying programs or applications. Sbin is for system binaries, which are the programs that make up your system. Dev contains references to your devices: hard drive, keyboard, mouse and so on. Etc contains your system configuration files.



```
pi@raspberrypi ~ $ ls /bin
bash      bzfgrep       chgrp      dash           domainname  fgconsole
bunzip2   bzgrep        chmod      date           dumpkeys    fgrep
bzcat     bzip2         chown      dd             echo        findmnt
bzcmp     bzip2recover  chvt       df             ed          fuser
bzdiff    bzless        con2fbmap  dir            egrep       fusermount
bzegrep   bzmore        cp         dmesg          false       grep
bzexe     cat           cpio       dnsdomainname  fbset       gunzip
pi@raspberrypi ~ $
```

STEP 4 Lib is a directory that contains libraries of code that are referred to by other programs (different programs share files in Lib). "Var" is short for various, which is mostly files used by the system, but you may need to work with items here. Finally there is a directory called "tmp", which is for temporary files; files placed here are on your system for the short term and can be deleted from the system.



```
pi@raspberrypi ~ $ ls /var
backups  cache  lib  local  lock  log  mail  opt  run  spool  swap  tmp
pi@raspberrypi ~ $
```

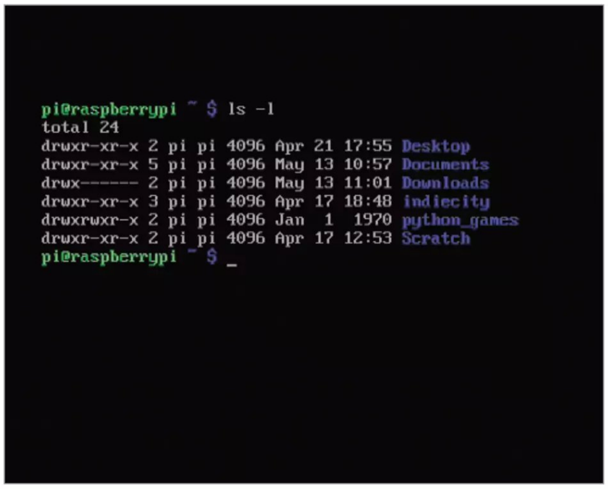
Creating and Deleting Files

Being able to create and delete a file is an everyday computing skill. However, when using the Linux Terminal, there's an element of care required, chiefly because any deleted files aren't placed in the system recycle bin.

CREATING FILES

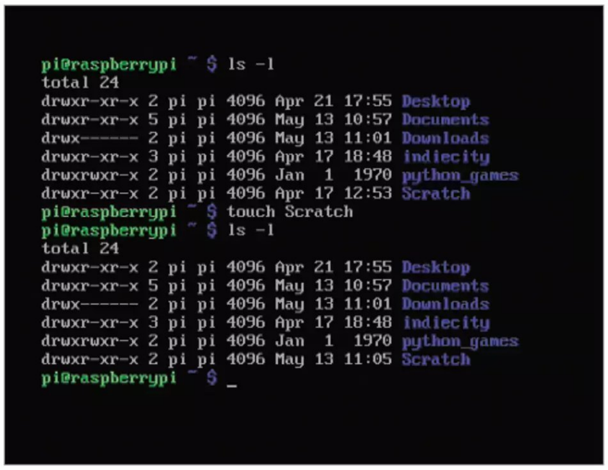
Once you learn to recognise the files and directories that make up Raspbian OS, it's time to discover how to make your own. Knowing how to make, edit and delete files and directories is essential if you want to make your own projects.

STEP 1 We're going to create a file using a command called Touch. Touch is an interesting command that reaches out to a file, or directory, and updates it (this changes the system time as if you'd just opened the file). You can see Touch in access using "ls –l" and checking the time next to a directory (such as Scratch).



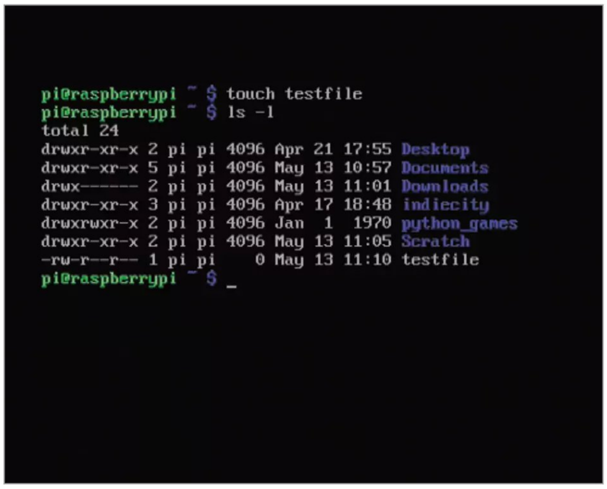
```
pi@raspberrypi ~ $ ls -l
total 24
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 5 pi pi 4096 May 13 10:57 Documents
drwx------ 2 pi pi 4096 May 13 11:01 Downloads
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 Apr 17 12:53 Scratch
pi@raspberrypi ~ $ _
```

STEP 2 Now enter: `touch Scratch` and `ls -l` again and notice that the time has changed. It now matches the current time. You might be wondering what this has to do with creating files or directories. Touch has a second, more popular, use, which is to create files.



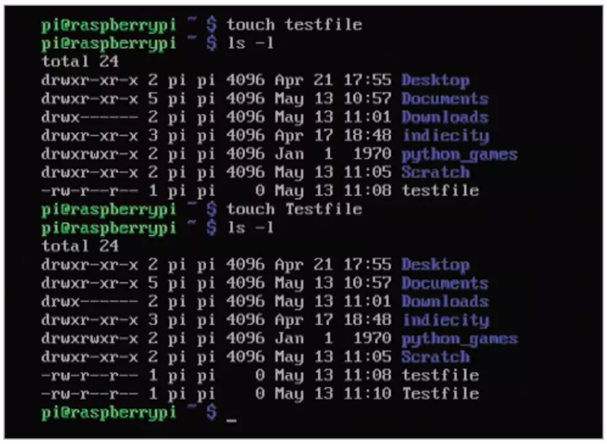
```
pi@raspberrypi ~ $ ls -l
total 24
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 5 pi pi 4096 May 13 10:57 Documents
drwx------ 2 pi pi 4096 May 13 11:01 Downloads
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 Apr 17 12:53 Scratch
pi@raspberrypi ~ $ touch Scratch
pi@raspberrypi ~ $ ls -l
total 24
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 5 pi pi 4096 May 13 10:57 Documents
drwx------ 2 pi pi 4096 May 13 11:01 Downloads
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 May 13 11:05 Scratch
pi@raspberrypi ~ $ _
```

STEP 3 If you try to touch a file that doesn't exist, you create a blank file with that name. Try it now. Type `touch testfile` and `ls -l` to view the files. You'll now have a new file in your home directory called "testfile". Notice that the size of the file is 0, because it has nothing in it.



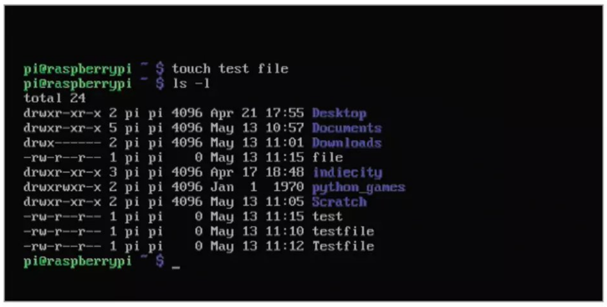
```
pi@raspberrypi ~ $ touch testfile
pi@raspberrypi ~ $ ls -l
total 24
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 5 pi pi 4096 May 13 10:57 Documents
drwx------ 2 pi pi 4096 May 13 11:01 Downloads
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 May 13 11:05 Scratch
-rw-r--r-- 1 pi pi    0 May 13 11:10 testfile
pi@raspberrypi ~ $ _
```

STEP 4 A quick word about file names: remember that Linux is case sensitive, so if you now enter: `touch Testfile` (with a capital T), it doesn't update 'testfile'; instead, it creates a second file called 'Testfile'. Enter: `ls -l` to see both files. This is confusing, so most people stick with using lowercase letters at all times.



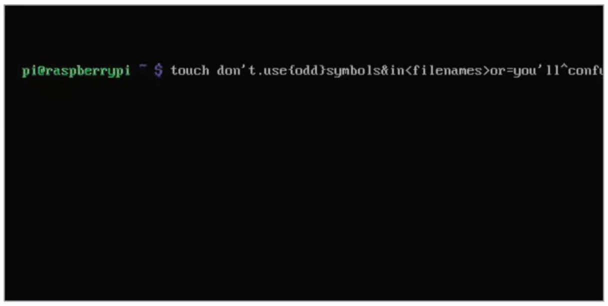
```
pi@raspberrypi ~ $ touch testfile
pi@raspberrypi ~ $ ls -l
total 24
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 5 pi pi 4096 May 13 10:57 Documents
drwx------ 2 pi pi 4096 May 13 11:01 Downloads
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 May 13 11:05 Scratch
-rw-r--r-- 1 pi pi    0 May 13 11:08 testfile
pi@raspberrypi ~ $ touch Testfile
pi@raspberrypi ~ $ ls -l
total 24
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 5 pi pi 4096 May 13 10:57 Documents
drwx------ 2 pi pi 4096 May 13 11:01 Downloads
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 May 13 11:05 Scratch
-rw-r--r-- 1 pi pi    0 May 13 11:08 testfile
-rw-r--r-- 1 pi pi    0 May 13 11:10 Testfile
pi@raspberrypi ~ $ _
```

STEP 5 Another important thing to know is never to use a space in your file names. If you try to enter: `touch test file`, you create a document called "test" and another called "file". Technically there are ways to create files containing a space but you should always use an underscore character ("_") instead of a space, such as "touch test_file".



```
pi@raspberrypi ~ $ touch test file
pi@raspberrypi ~ $ ls -l
total 24
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 5 pi pi 4096 May 13 10:57 Documents
drwx------ 2 pi pi 4096 May 13 11:01 Downloads
-rw-r--r-- 1 pi pi    0 May 13 11:15 file
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 May 13 11:05 Scratch
-rw-r--r-- 1 pi pi    0 May 13 11:15 test
-rw-r--r-- 1 pi pi    0 May 13 11:10 testfile
-rw-r--r-- 1 pi pi    0 May 13 11:12 Testfile
pi@raspberrypi ~ $ _
```

STEP 6 Here are some other files names to avoid: #%&{}\<>*?/$!'":@+`|=. The full stop (.) is used to create an extension to a file; usually used to indicate a file type, such as textfile.txt or compressedfile.zip, and starting a file with a full stop makes it invisible. Don't use a full stop in place of a space though; stick to underscores.

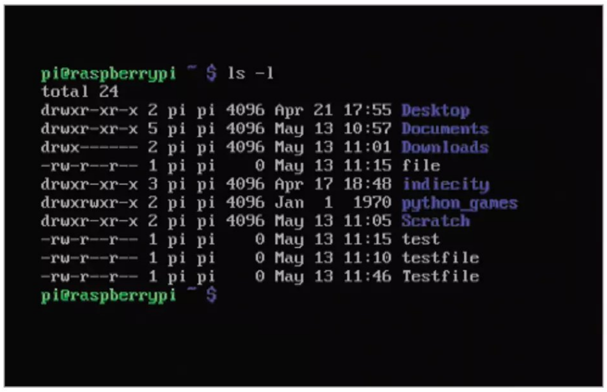


```
pi@raspberrypi ~ $ touch don't.use{odd}symbols&in<filenames>or=you'll^confu
```

REMOVING FILES

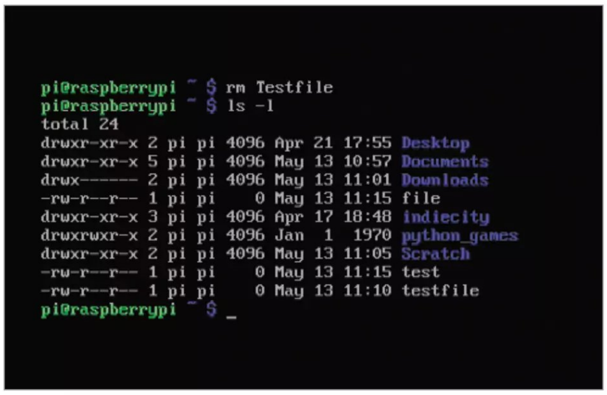
We've created some files that we don't want, so how do we go about removing them? It turns out that deleting files in your Raspberry Pi is really easy, which may be a problem, so be careful.

STEP 1 Enter: `ls -l` to view the files in your home directory. If you've followed the steps before then you should have three files: "test", "testfile", and "Testfile". We're going to get rid of these items because they were created as an example.



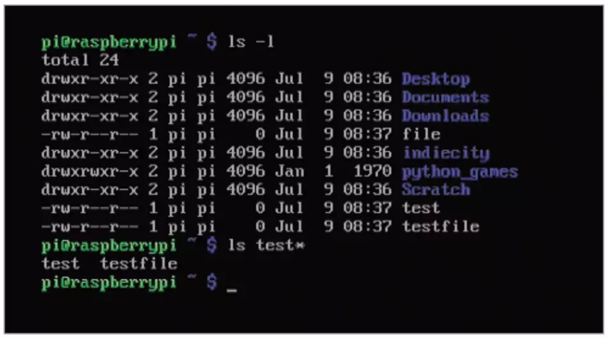
```
pi@raspberrypi ~ $ ls -l
total 24
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 5 pi pi 4096 May 13 10:57 Documents
drwx------ 2 pi pi 4096 May 13 11:01 Downloads
-rw-r--r-- 1 pi pi    0 May 13 11:15 file
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 May 13 11:05 Scratch
-rw-r--r-- 1 pi pi    0 May 13 11:15 test
-rw-r--r-- 1 pi pi    0 May 13 11:10 testfile
-rw-r--r-- 1 pi pi    0 May 13 11:46 Testfile
pi@raspberrypi ~ $
```

STEP 2 **To get rid of files you use the "rm" command. Enter: `rm Testfile` to delete the file called "Testfile" (with the uppercase "t"). Enter: `ls -l` and you'll find it's gone. Where is it? It's not in the Trash or Recycle Bin, like on a Mac or Windows PC. It's deleted completely and cannot be recovered. Bear this in mind and always think before deleting files.**



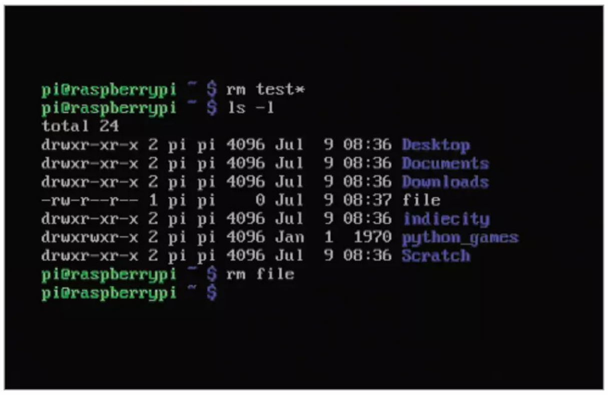
```
pi@raspberrypi ~ $ rm Testfile
pi@raspberrypi ~ $ ls -l
total 24
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 5 pi pi 4096 May 13 10:57 Documents
drwx------ 2 pi pi 4096 May 13 11:01 Downloads
-rw-r--r-- 1 pi pi    0 May 13 11:15 file
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 May 13 11:05 Scratch
-rw-r--r-- 1 pi pi    0 May 13 11:15 test
-rw-r--r-- 1 pi pi    0 May 13 11:10 testfile
pi@raspberrypi ~ $ _
```

STEP 3 We're going to use a wildcard (*) to delete our next two files, but again this is something you really need to do with care. First use "ls" to list the files and make sure it's the one you want to delete. Enter: `ls test*` to view files that match the word "test" and any other characters. The "*" character is called a "wildcard" and it means any characters here.



```
pi@raspberrypi ~ $ ls -l
total 24
drwxr-xr-x 2 pi pi 4096 Jul  9 08:36 Desktop
drwxr-xr-x 2 pi pi 4096 Jul  9 08:36 Documents
drwxr-xr-x 2 pi pi 4096 Jul  9 08:36 Downloads
-rw-r--r-- 1 pi pi    0 Jul  9 08:37 file
drwxr-xr-x 2 pi pi 4096 Jul  9 08:36 indiecity
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 Jul  9 08:36 Scratch
-rw-r--r-- 1 pi pi    0 Jul  9 08:37 test
-rw-r--r-- 1 pi pi    0 Jul  9 08:37 testfile
pi@raspberrypi ~ $ ls test*
test  testfile
pi@raspberrypi ~ $ _
```

STEP 4 We see that "ls test*" matches two files: "test" and "testfile", but not the file called "file". That's because it didn't match the "test" part of "test*". Check carefully over groups of files you want to remove (remember you can't recover them) and replace the "ls" with "rm". Enter: `rm test*` to remove both files. Finally enter: `rm file` to get rid of the confusing file.



```
pi@raspberrypi ~ $ rm test*
pi@raspberrypi ~ $ ls -l
total 24
drwxr-xr-x 2 pi pi 4096 Jul  9 08:36 Desktop
drwxr-xr-x 2 pi pi 4096 Jul  9 08:36 Documents
drwxr-xr-x 2 pi pi 4096 Jul  9 08:36 Downloads
-rw-r--r-- 1 pi pi    0 Jul  9 08:37 file
drwxr-xr-x 2 pi pi 4096 Jul  9 08:36 indiecity
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 Jul  9 08:36 Scratch
pi@raspberrypi ~ $ rm file
pi@raspberrypi ~ $
```

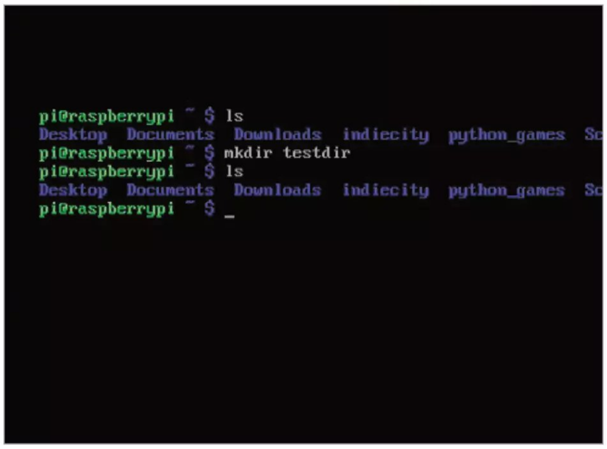
Create and Remove Directories

Creating, moving and deleting directories isn't as easy in the Terminal as it is within a desktop interface. You need to tell Linux to move the directories inside other directories, a process known as recursion. Sounds complex but you should quickly get the hang of it.

MANAGING FILES AND DIRECTORIES

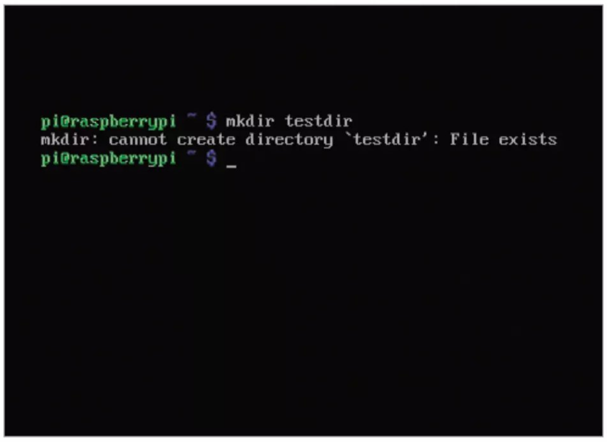
Now that you know how to create files, you'll want to learn how to make directories, which are the same thing as folders, as well as move items around. If you are more used to working with a desktop interface, this can take a bit of getting used to.

STEP 1 Enter: `ls` to quickly view all the directories currently in in the home location. Directories are created using the "mkdir" command (make directory). Enter: `mkdir testdir` to create a new directory in your home directory. Enter: `ls` again to see it.



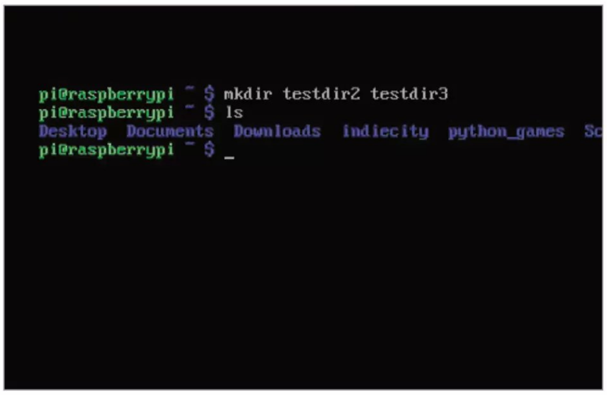
```
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  python_games  Sc
pi@raspberrypi ~ $ mkdir testdir
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  python_games  Sc
pi@raspberrypi ~ $ _
```

STEP 2 The "mkdir" command is different to touch, in that it doesn't update the timestamp if you use it with a directory that already exists. Enter: `mkdir testdir` again and you'll get the error "mkdir: cannot create directory 'testdir: File exists".



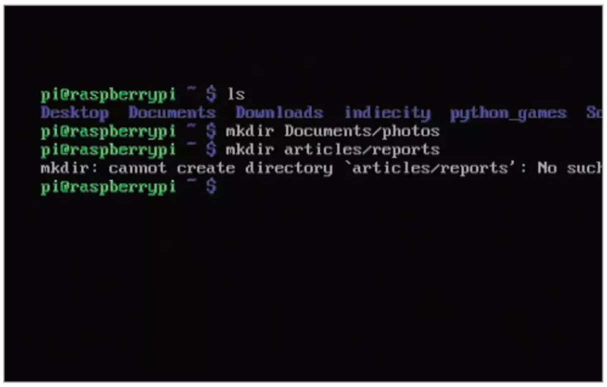
```
pi@raspberrypi ~ $ mkdir testdir
mkdir: cannot create directory `testdir': File exists
pi@raspberrypi ~ $ _
```

STEP 3 Like touch, you can create multiple directories at once with the mkdir command. Enter: `mkdir testdir2 testdir3` and enter: `ls`. You'll now find several directories called testdir. Also, like files, you should know this means you can't (and really shouldn't) create directories with spaces. As with files, use an underscore ("_") character instead of a space.



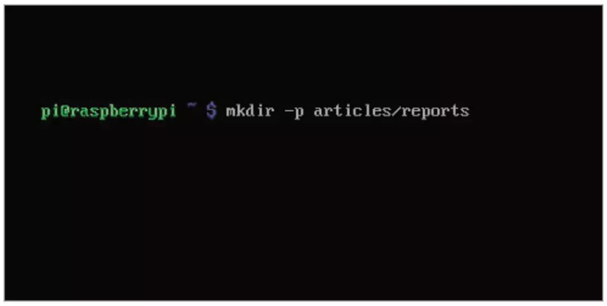
```
pi@raspberrypi ~ $ mkdir testdir2 testdir3
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  python_games  Sc
pi@raspberrypi ~ $ _
```

STEP 4 You can create directories inside of each other using the directory path. Enter: `mkdir Documents/photos` to create a new directory called "photos" inside your documents directory. The directory has to already exist, though, try to enter: `mkdir articles/reports` and you'll get an error because there is no articles directory.



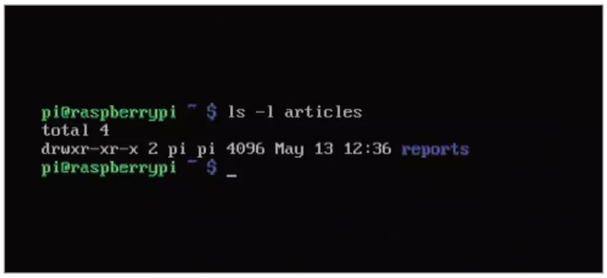
```
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  python_games  Sc
pi@raspberrypi ~ $ mkdir Documents/photos
pi@raspberrypi ~ $ mkdir articles/reports
mkdir: cannot create directory `articles/reports': No such
pi@raspberrypi ~ $
```

STEP 5 To create a directory path you need to pass in the “p” option to mkdir (which stands for “parents”). Options, if you remember, come after the command and start with a ‘-’. So enter: `mkdir -p articles/reports`. Enter: `ls` to view the articles directory, or “ls articles” to view the reports directory sitting inside.



```
pi@raspberrypi ~ $ mkdir -p articles/reports
```

STEP 6 Now you’re starting to get a bit more advanced, we’re going to just reiterate something. In Linux the command structure is always: command, option and argument, in that order. The command is the function, next are the options (typically single letters starting with “-”) and finally the argument (often a file, or directory structure). It’s always command, option then argument.

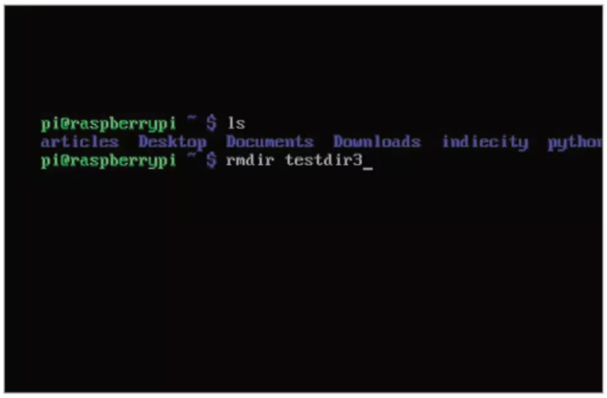


```
pi@raspberrypi ~ $ ls -l articles
total 4
drwxr-xr-x 2 pi pi 4096 May 13 12:36 reports
pi@raspberrypi ~ $ _
```

GETTING RID OF DIRECTORIES

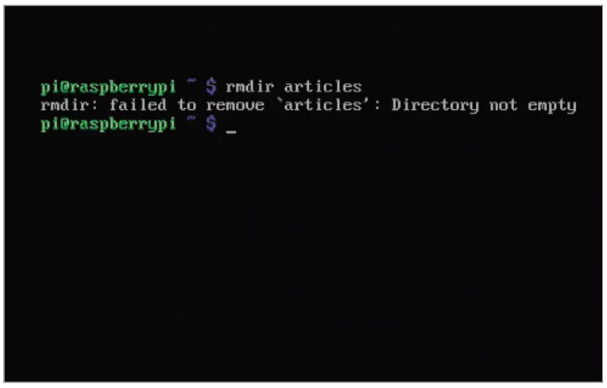
Deleting directories is pretty easy in Linux, along with files, and this can be a problem. It’s too easy to delete entire directories containing files and these are instantly removed, not sent to a trash directory. Tread carefully.

STEP 1 We’re going to remove one of the directories we created earlier using the “rmdir” command. Enter: `ls` to view the files and directories in the current directory. We’ll start by getting rid of one of the test directories. Enter: `rmdir testdir3` and `ls` again to confirm the directory has been removed.



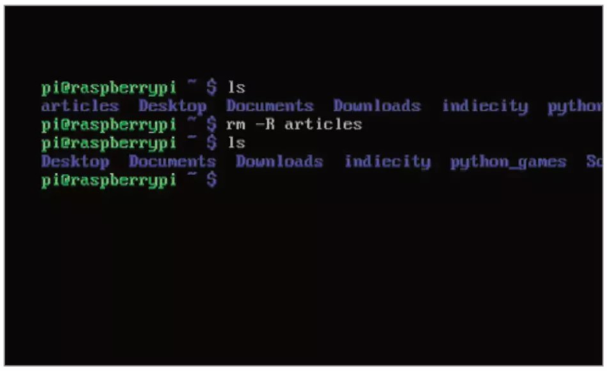
```
pi@raspberrypi ~ $ ls
articles  Desktop  Documents  Downloads  indiecity  pytho
pi@raspberrypi ~ $ rmdir testdir3_
```

STEP 2 Now we’ll try to get rid of the articles directory (containing the reports directory). Enter: `rmdir articles` and press return. You’ll get an error saying “rmdir: failed to remove ‘articles’: Directory not empty”. This is a puzzler; the rmdir command only removes directories that having nothing in them (no files or other directories).



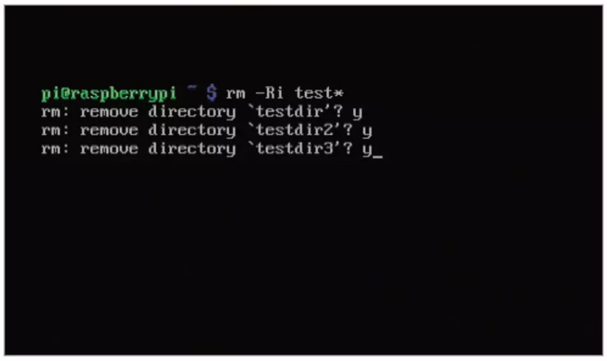
```
pi@raspberrypi ~ $ rmdir articles
rmdir: failed to remove `articles': Directory not empty
pi@raspberrypi ~ $ _
```

STEP 3 To delete a directory containing files or other directories, you return to the “rm” command used to remove files, only now we need to use the “-R” option (which stands for “recursive”.) Using “rm –R” removes all the files and directories to whatever you point it at. Enter: `rm -R articles` to remove the articles directory.



```
pi@raspberrypi ~ $ ls
articles  Desktop  Documents  Downloads  indiecity  pytho
pi@raspberrypi ~ $ rm -R articles
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  python_games  Sc
pi@raspberrypi ~ $
```

STEP 4 As with multiple files, you can delete multiple directories inside the same directory using the “rm” command with the wildcard character (`*`). This should be done with care though so use the `-I` option (which stands for “interactive”). This will prompt you before each deletion. Enter: `rm -Ri test*` and press `Y` and `return` to each prompt. It’s a good idea to use the `-i` option whenever using the `rm` command.



```
pi@raspberrypi ~ $ rm -Ri test*
rm: remove directory `testdir'? y
rm: remove directory `testdir2'? y
rm: remove directory `testdir3'? y_
```

Copying, Moving and Renaming Files

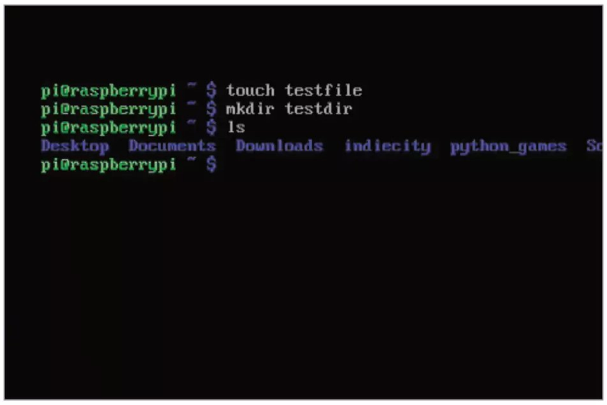
Taking command of the Terminal is essential when learning how your Raspberry Pi's operating system works. The copying, moving and renaming of files is equally important, as you'll be doing a lot of this throughout your Pi projects.

USING THE MOVE COMMAND

In Linux, renaming a file is simply moving it from one name to another and copying a file is moving it without deleting the original. Don't panic, it's quite easy to master.

STEP 1

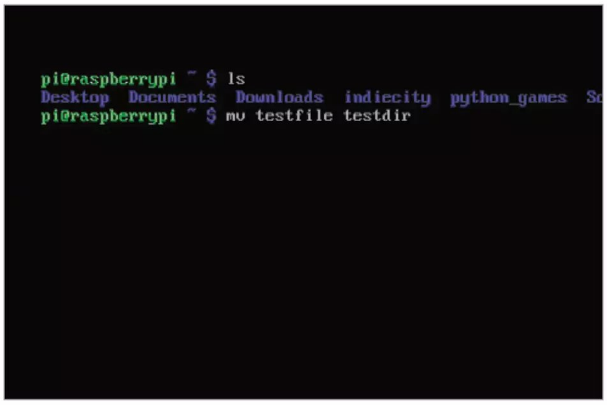
Before we can move anything around, we need to have a few test items in our home directory. Enter: `touch testfile` and `mkdir testdir` to create a test file and test directory in your home directory. Enter: `ls` to check that they are both present.



```
pi@raspberrypi ~ $ touch testfile
pi@raspberrypi ~ $ mkdir testdir
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  python_games  Sc
pi@raspberrypi ~ $
```

STEP 2

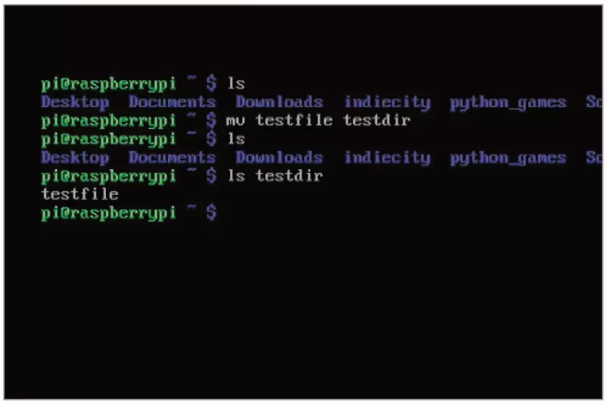
Files and directories are moved using the mv command. This is different to the commands we've looked at so far because it has two arguments (remember Linux command line is command, option, argument). The first argument is the source (the file or directory to be moved) and the second is the destination.



```
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  python_games  Sc
pi@raspberrypi ~ $ mv testfile testdir
```

STEP 3

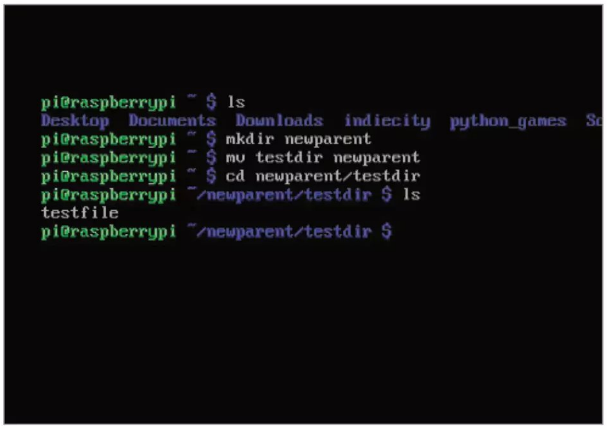
Enter: `mv testfile testdir` and press return to move the testfile document into the testdir directory. Enter: `ls` to see that it's no longer in the home directory, and `ls testdir` to see the testfile now sitting in the testdir directory. Now enter: `mkdir newparent` to create a new directory.



```
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  python_games  Sc
pi@raspberrypi ~ $ mv testfile testdir
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  python_games  Sc
pi@raspberrypi ~ $ ls testdir
testfile
pi@raspberrypi ~ $
```

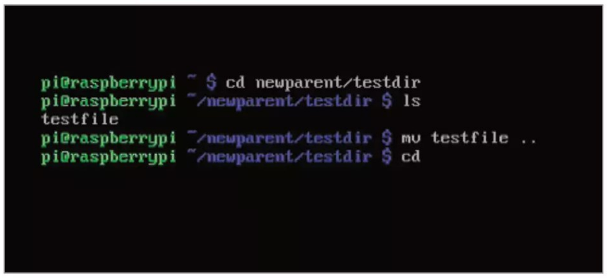
STEP 4

Directories with files are moved in the same way. Enter: `mv testdir newparent` to move the testdir directory inside the newparent directory. Let's move into the directory to find the file. Enter: `cd /newparent/testdir` and enter: `ls` to view the testfile sitting inside the directory.



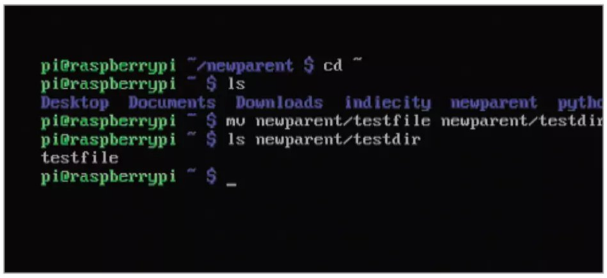
```
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  python_games  Sc
pi@raspberrypi ~ $ mkdir newparent
pi@raspberrypi ~ $ mv testdir newparent
pi@raspberrypi ~ $ cd newparent/testdir
pi@raspberrypi ~/newparent/testdir $ ls
testfile
pi@raspberrypi ~/newparent/testdir $
```

STEP 5 Files and directories can be moved up using the double dot ("..") as an argument. Enter: `ls -la` to view your testfile and the single and double dot files. The single dot is the current directory and the double dot is the parent directory. Enter: `mv testfile ..` to move the testfile up into the newparent directory. Enter: `cd ..` to move up to the parent directory.



```
pi@raspberrypi ~ $ cd newparent/testdir
pi@raspberrypi ~/newparent/testdir $ ls
testfile
pi@raspberrypi ~/newparent/testdir $ mv testfile ..
pi@raspberrypi ~/newparent/testdir $ cd
```

STEP 6 You can also move files using longer paths. Enter: `cd ~` to return to the home directory and `mv newparent/testfile newparent/testdir/testfile` to move the testfile from its current location back inside the testdir directory. Enter: `ls newparent/testdir` to view the file back in its current directory.

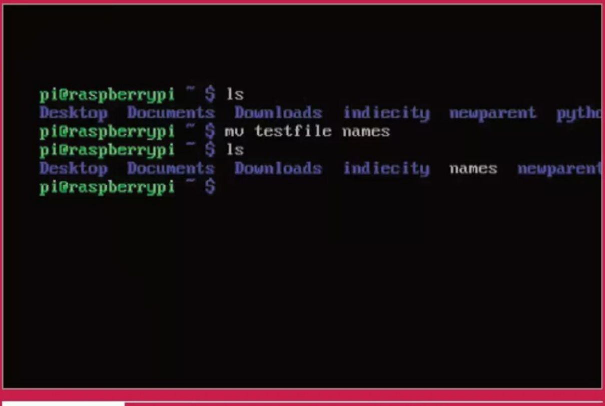


```
pi@raspberrypi ~/newparent $ cd ~
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  newparent  pytho
pi@raspberrypi ~ $ mv newparent/testfile newparent/testdir
pi@raspberrypi ~ $ ls newparent/testdir
testfile
pi@raspberrypi ~ $ _
```

RENAMING FILES AND DIRECTORIES

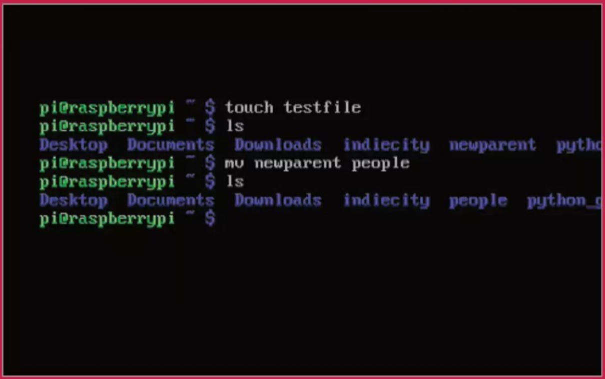
The mv command isn't used just to move files; it also serves the purpose of renaming files (effectively it moves it from its old name to a new name). Let's see how to use mv to rename items.

STEP 1 Let's start by making a new test file called "names". Enter: `touch testfile` and then `ls` to make sure the testfile is present. We're going to turn this into a file that contains the names of some people. So let's call it something more appropriate, like "names".



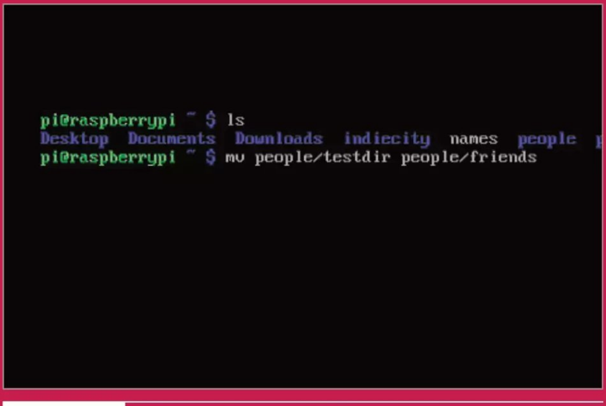
```
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  newparent  pytho
pi@raspberrypi ~ $ mv testfile names
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  names  newparent
pi@raspberrypi ~ $
```

STEP 2 Enter: `mv testfile names` and `ls`. Now we can see the new "names" file in our directory. The mv command can also be used to rename directories. We should still have our newparent directory in our home directory. Enter: `mv newparent people` to rename the newparent directory. Enter: `ls` to view it.



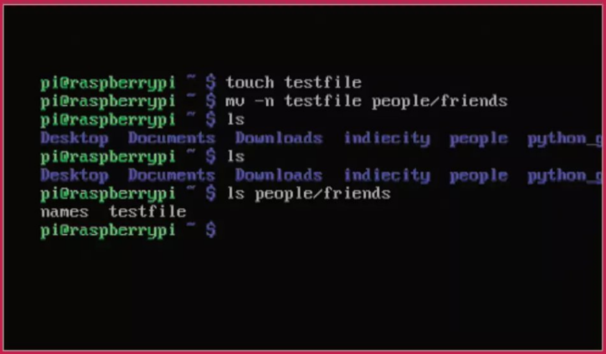
```
pi@raspberrypi ~ $ touch testfile
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  newparent  pytho
pi@raspberrypi ~ $ mv newparent people
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  people  python_g
pi@raspberrypi ~ $
```

STEP 3 You can rename directories inside other directories using paths. Let's rename the testdir directory, which is now inside the people directory. Enter: `mv names/testdir names/friends`. Now enter: `mv names people/friends` to move the names file inside the friends directory.



```
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  names  people  p
pi@raspberrypi ~ $ mv people/testdir people/friends
```

STEP 4 It is easy to overwrite files using the mv command, so if you have files with the same name use the "-n" option, which stands for "no overwrite". Enter: `touch testfile` to create a new file and `mv -n testfile people/friends`. There's no error report though, enter: `ls` and you'll find testfile still there.



```
pi@raspberrypi ~ $ touch testfile
pi@raspberrypi ~ $ mv -n testfile people/friends
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  people  python_g
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  people  python_g
pi@raspberrypi ~ $ ls people/friends
names  testfile
pi@raspberrypi ~ $
```

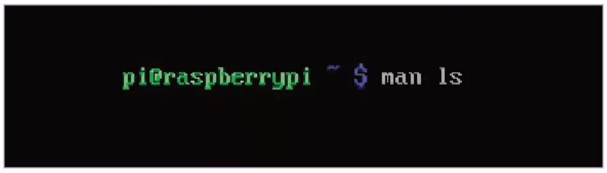
Using the Man Pages

Linux comes with man (manual) pages that explain each command and show you all the options you can use. Once you get the hang of reading the man pages, you'll be able to find and do just about anything in Linux.

HEY, MAN!

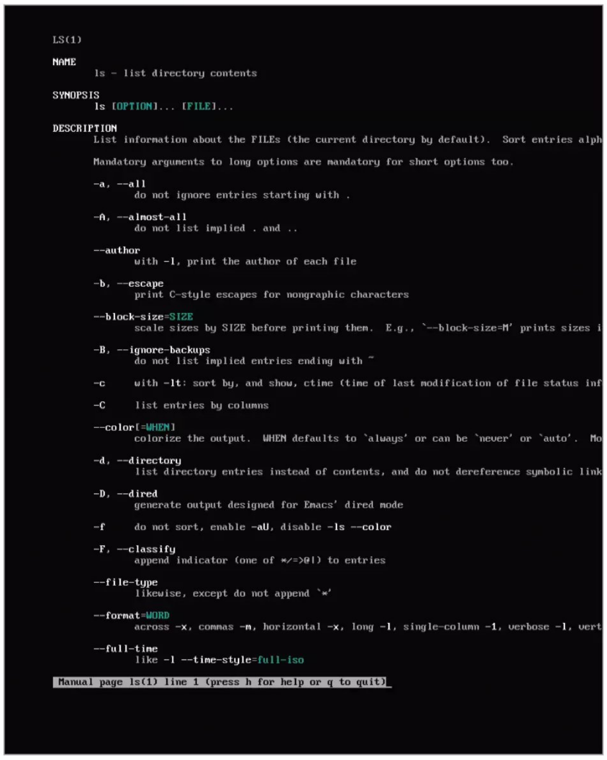
The man pages are one of the best features of Linux, and as a built-in tool it's invaluable for both beginner and senior level Linux administrators. Let's see how it works

STEP 1 Linux has a built-in manual, known as man for short. Using the man command you can obtain information on all the Linux commands we've talked about. Simply enter: `man` and the name of the command you want to learn more about. Start by entering: `man ls` in the command line.



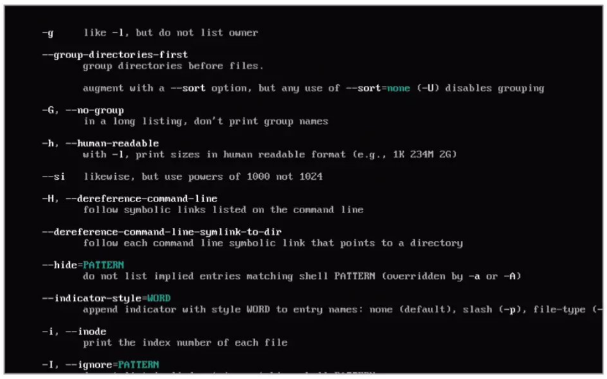
```
pi@raspberrypi ~ $ man ls
```

STEP 2 The man pages are a bit more detailed than you might be used to. First you have a name, which tells you what the command is called; in this case "list directory contents" and then the synopsis shows you how it works. In this case: "ls [OPTION].. [FILE..]". So you enter: `ls` followed by options (such as `-la`) and the file or directory to list.



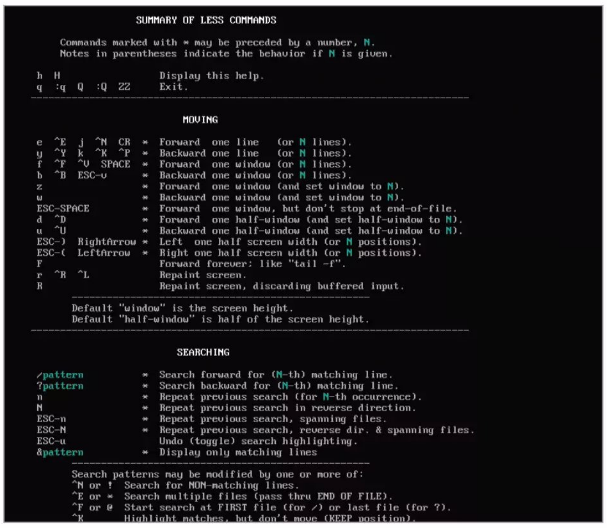
```
LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List information about the FILEs (the current directory by default).  Sort entries alph

       Mandatory arguments to long options are mandatory for short options too.

       -a, --all
              do not ignore entries starting with .

       -A, --almost-all
              do not list implied . and ..

       --author
              with -l, print the author of each file

       -b, --escape
              print C-style escapes for nongraphic characters

       --block-size=SIZE
              scale sizes by SIZE before printing them.  E.g., `--block-size=M' prints sizes i

       -B, --ignore-backups
              do not list implied entries ending with ~

       -c     with -lt: sort by, and show, ctime (time of last modification of file status inf

       -C     list entries by columns

       --color[=WHEN]
              colorize the output.  WHEN defaults to `always' or can be `never' or `auto'.  Mo

       -d, --directory
              list directory entries instead of contents, and do not dereference symbolic link

       -D, --dired
              generate output designed for Emacs' dired mode

       -f     do not sort, enable -aU, disable -ls --color

       -F, --classify
              append indicator (one of */=>@|) to entries

       --file-type
              likewise, except do not append `*'

       --format=WORD
              across -x, commas -m, horizontal -x, long -l, single-column -1, verbose -l, vert

       --full-time
              like -l --time-style=full-iso

 Manual page ls(1) line 1 (press h for help or q to quit)
```

STEP 3 Most commands are pretty easy to figure out how to use, so what you spend most of the time in the man pages is looking under the Description. Here you will see all the options and the letters used to activate them. Most man pages are longer than a single page, so press any key, such as the space bar, to move to the next page of content.



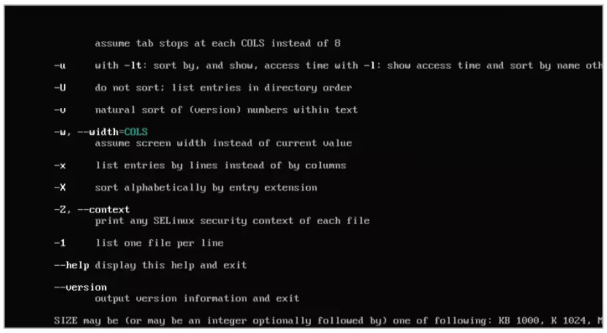
```
-g     like -l, but do not list owner

--group-directories-first
       group directories before files.

       augment with a --sort option, but any use of --sort=none (-U) disables grouping

-G, --no-group
       in a long listing, don't print group names

-h, --human-readable
       with -l, print sizes in human readable format (e.g., 1K 234M 2G)

--si   likewise, but use powers of 1000 not 1024

-H, --dereference-command-line
       follow symbolic links listed on the command line

--dereference-command-line-symlink-to-dir
       follow each command line symbolic link that points to a directory

--hide=PATTERN
       do not list implied entries matching shell PATTERN (overridden by -a or -A)

--indicator-style=WORD
       append indicator with style WORD to entry names: none (default), slash (-p), file-type (-

-i, --inode
       print the index number of each file

-I, --ignore=PATTERN
```

STEP 4 Press the H key while looking at a man page to view the commands you can use to control the view. This is called the Summary of Less Commands (the less command is something we'll come to when we look at editing text). For now realise that you can move back and forward with Z and W. Press Q to quit this help screen and return to the man page.

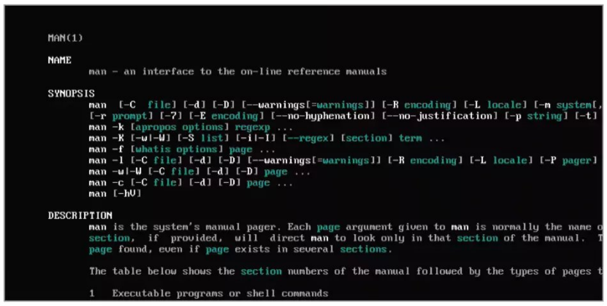


```
                    SUMMARY OF LESS COMMANDS

      Commands marked with * may be preceded by a number, N.
      Notes in parentheses indicate the behavior if N is given.

  h  H                 Display this help.
  q  :q  Q  :Q  ZZ     Exit.
 ---------------------------------------------------------------------------

                          MOVING

  e  ^E  j  ^N  CR  *  Forward  one line   (or N lines).
  y  ^Y  k  ^K  ^P  *  Backward one line   (or N lines).
  f  ^F  ^V  SPACE  *  Forward  one window (or N lines).
  b  ^B  ESC-v      *  Backward one window (or N lines).
  z                 *  Forward  one window (and set window to N).
  w                 *  Backward one window (and set window to N).
  ESC-SPACE         *  Forward  one window, but don't stop at end-of-file.
  d  ^D             *  Forward  one half-window (and set half-window to N).
  u  ^U             *  Backward one half-window (and set half-window to N).
  ESC-)  RightArrow *  Left  one half screen width (or N positions).
  ESC-(  LeftArrow  *  Right one half screen width (or N positions).
  F                    Forward forever; like "tail -f".
  r  ^R  ^L            Repaint screen.
  R                    Repaint screen, discarding buffered input.
        ---------------------------------------------------
        Default "window" is the screen height.
        Default "half-window" is half of the screen height.
 ---------------------------------------------------------------------------

                         SEARCHING

  /pattern          *  Search forward for (N-th) matching line.
  ?pattern          *  Search backward for (N-th) matching line.
  n                 *  Repeat previous search (for N-th occurrence).
  N                 *  Repeat previous search in reverse direction.
  ESC-n             *  Repeat previous search, spanning files.
  ESC-N             *  Repeat previous search, reverse dir. & spanning files.
  ESC-u                Undo (toggle) search highlighting.
  &pattern          *  Display only matching lines
        ---------------------------------------------------
        Search patterns may be modified by one or more of:
        ^N or !  Search for NON-matching lines.
        ^E or *  Search multiple files (pass thru END OF FILE).
        ^F or @  Start search at FIRST file (for /) or last file (for ?).
        ^K       Highlight matches, but don't move (KEEP position).
```

STEP 5 Scroll to the bottom of the man page to discover more information. Typically you will find the author's name and information on reporting bugs, including web links that can be useful for more information. Press Q to exit the man page and return to the command line.


```
              assume tab stops at each COLS instead of 8

       -u     with -lt: sort by, and show, access time with -l: show access time and sort by name oth

       -U     do not sort; list entries in directory order

       -v     natural sort of (version) numbers within text

       -w, --width=COLS
              assume screen width instead of current value

       -x     list entries by lines instead of by columns

       -X     sort alphabetically by entry extension

       -Z, --context
              print any SELinux security context of each file

       -1     list one file per line

       --help display this help and exit

       --version
              output version information and exit

       SIZE may be (or may be an integer optionally followed by) one of following: KB 1000, K 1024, M
```

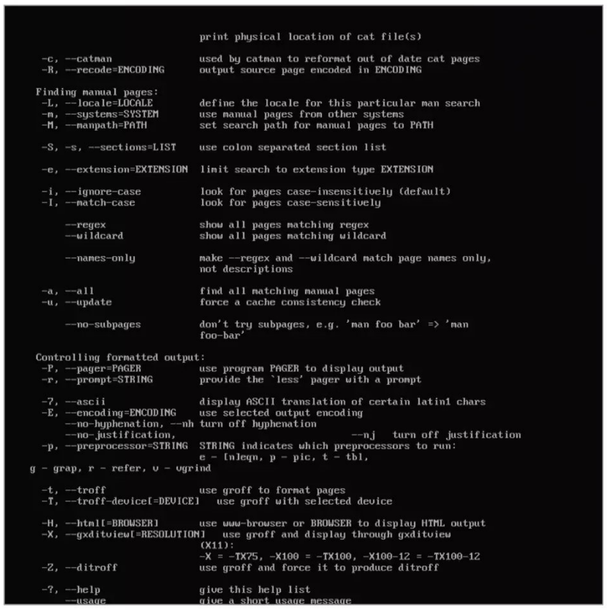
STEP 6 The man command can be used for just about every command you use in Linux. You can even enter: man man to get information on using the man tool. From now on, whenever you come across a new command in this book, such as "nano" or "chmod", take time to enter: `man nano` or `man chmod` and read the instructions.


```
MAN(1)

NAME
       man - an interface to the on-line reference manuals

SYNOPSIS
       man  [-C  file] [-d] [-D] [--warnings[=warnings]] [-R encoding] [-L locale] [-m system[,
       [-r prompt] [-7] [-E encoding] [--no-hyphenation] [--no-justification] [-p string] [-t]
       man -k [apropos options] regexp ...
       man -K [-w|-W] [-S list] [-i|-I] [--regex] [section] term ...
       man -f [whatis options] page ...
       man -l [-C file] [-d] [-D] [--warnings[=warnings]] [-R encoding] [-L locale] [-P pager]
       man -w|-W [-C file] [-d] [-D] page ...
       man -c [-C file] [-d] [-D] page ...
       man [-hV]

DESCRIPTION
       man is the system's manual pager. Each page argument given to man is normally the name o
       section,  if  provided,  will  direct man to look only in that section of the manual.  T
       page found, even if page exists in several sections.

       The table below shows the section numbers of the manual followed by the types of pages t

       1   Executable programs or shell commands
```

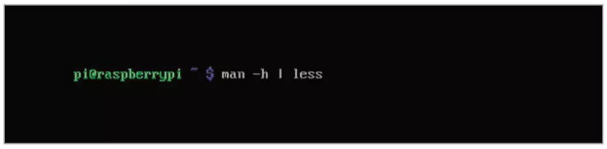
USING MAN OPTIONS

Because man doesn't change anything, like mv or mkdir, it is tempting not to see it as a command. But it is, and like all other commands it has options. These can be very handy to learn.

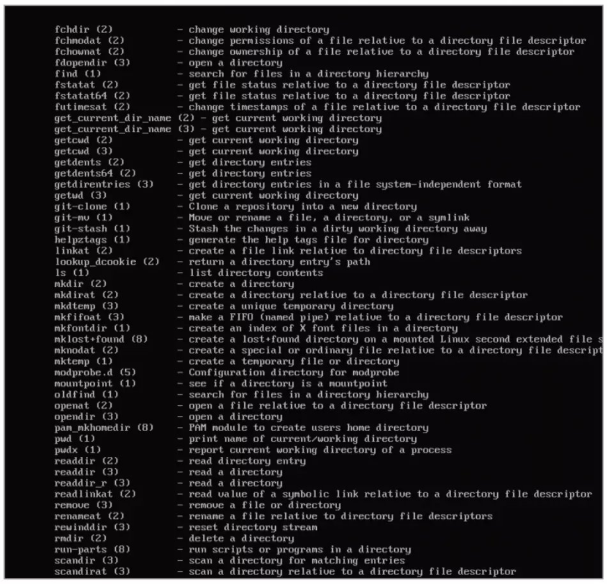
STEP 1 Entering: `man man` enables you to view some of the options, but sometimes you'll just want a quick overview. Fortunately man has a built-in help option that quickly lists the options. Press Q if you're in a man page and enter: `man -h` at the command line.


```
                             print physical location of cat file(s)

  -c, --catman               used by catman to reformat out of date cat pages
  -R, --recode=ENCODING      output source page encoded in ENCODING

 Finding manual pages:
  -L, --locale=LOCALE        define the locale for this particular man search
  -m, --systems=SYSTEM       use manual pages from other systems
  -M, --manpath=PATH         set search path for manual pages to PATH

  -S, -s, --sections=LIST    use colon separated section list

  -e, --extension=EXTENSION  limit search to extension type EXTENSION

  -i, --ignore-case          look for pages case-insensitively (default)
  -I, --match-case           look for pages case-sensitively

      --regex                show all pages matching regex
      --wildcard             show all pages matching wildcard

      --names-only           make --regex and --wildcard match page names only,
                             not descriptions

  -a, --all                  find all matching manual pages
  -u, --update               force a cache consistency check

      --no-subpages          don't try subpages, e.g. 'man foo bar' => 'man
                             foo-bar'

 Controlling formatted output:
  -P, --pager=PAGER          use program PAGER to display output
  -r, --prompt=STRING        provide the `less' pager with a prompt

  -7, --ascii                display ASCII translation of certain latin1 chars
  -E, --encoding=ENCODING    use selected output encoding
      --no-hyphenation, --nh turn off hyphenation
      --no-justification,                              --nj   turn off justification
  -p, --preprocessor=STRING  STRING indicates which preprocessors to run:
                             e - [n]eqn, p - pic, t - tbl,
g - grap, r - refer, v - vgrind

  -t, --troff                use groff to format pages
  -T, --troff-device[=DEVICE]   use groff with selected device

  -H, --html[=BROWSER]       use www-browser or BROWSER to display HTML output
  -X, --gxditview[=RESOLUTION]   use groff and display through gxditview
                             (X11):
                             -X = -TX75, -X100 = -TX100, -X100-12 = -TX100-12
  -Z, --ditroff              use groff and force it to produce ditroff

  -?, --help                 give this help list
      --usage                give a short usage message
```

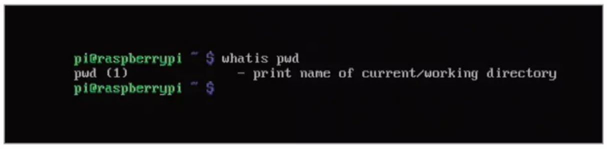
STEP 2 If you're fast you may have noticed the start of the text flew up off the page. This is because the "man –h" option doesn't use the less command by default (less is what enables you to move down text one screen at a time). We'll look into pipes ("|") later on, but for now just use "man -h | less" to read long text one page at a time.


```
pi@raspberrypi ~ $ man -h | less
```

STEP 3 One of the most powerful man options is the -k option, which is for "apropos". This enables you to search a wider range of man pages than the exact command. Enter: `man -k directory` to view all of the man pages relating to directories "(man -k directory | less" to view one page at a time). Here you'll find commands like "ls", "mkdir" and "cd" along with their description.


```
fchdir (2)           - change working directory
fchmodat (2)         - change permissions of a file relative to a directory file descriptor
fchownat (2)         - change ownership of a file relative to a directory file descriptor
fdopendir (3)        - open a directory
find (1)             - search for files in a directory hierarchy
fstatat (2)          - get file status relative to a directory file descriptor
fstatat64 (2)        - get file status relative to a directory file descriptor
futimesat (2)        - change timestamps of a file relative to a directory file descriptor
get_current_dir_name (2) - get current working directory
get_current_dir_name (3) - get current working directory
getcwd (2)           - get current working directory
getcwd (3)           - get current working directory
getdents (2)         - get directory entries
getdents64 (2)       - get directory entries
getdirentries (3)    - get directory entries in a file system-independent format
getwd (3)            - get current working directory
git-clone (1)        - Clone a repository into a new directory
git-mv (1)           - Move or rename a file, a directory, or a symlink
git-stash (1)        - Stash the changes in a dirty working directory away
helpztags (1)        - generate the help tags file for directory
linkat (2)           - create a file link relative to directory file descriptors
lookup_dcookie (2)   - return a directory entry's path
ls (1)               - list directory contents
mkdir (2)            - create a directory
mkdirat (2)          - create a directory relative to a directory file descriptor
mkdtemp (3)          - create a unique temporary directory
mkfifoat (3)         - make a FIFO (named pipe) relative to a directory file descriptor
mkfontdir (1)        - create an index of X font files in a directory
mklost+found (8)     - create a lost+found directory on a mounted Linux second extended file s
mknodat (2)          - create a special or ordinary file relative to a directory file descript
mktemp (1)           - create a temporary file or directory
modprobe.d (5)       - Configuration directory for modprobe
mountpoint (1)       - see if a directory is a mountpoint
oldfind (1)          - search for files in a directory hierarchy
openat (2)           - open a file relative to a directory file descriptor
opendir (3)          - open a directory
pam_mkhomedir (8)    - PAM module to create users home directory
pwd (1)              - print name of current/working directory
pwdx (1)             - report current working directory of a process
readdir (2)          - read directory entry
readdir (3)          - read a directory
readdir_r (3)        - read a directory
readlinkat (2)       - read value of a symbolic link relative to a directory file descriptor
remove (3)           - remove a file or directory
renameat (2)         - rename a file relative to directory file descriptors
rewinddir (3)        - reset directory stream
rmdir (2)            - delete a directory
run-parts (8)        - run scripts or programs in a directory
scandir (3)          - scan a directory for matching entries
scandirat (3)        - scan a directory relative to a directory file descriptor
```

STEP 4 Entering the man page for all the commands you come across can be a little long-winded, although ultimately productive. If you simply want to know what a command does you can read just the description using the "whatis" command. Enter: `whatis pwd` to read the description of the "pwd" command ("print name of current/working directory").


```
pi@raspberrypi ~ $ whatis pwd
pwd (1)              - print name of current/working directory
pi@raspberrypi ~ $
```

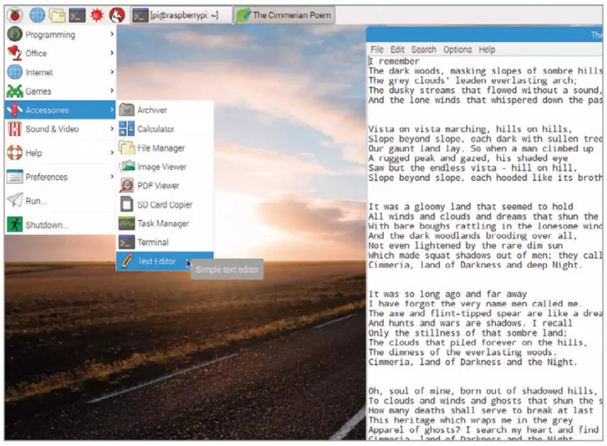
Editing Text Files

A text file in Linux can be anything from a simple set of instructions on how to use an app, to some complex Python, C++ or other programming language code. Text files can be used for scripting, automated executable files, as well as configuration files too.

THE JOY OF TEXT

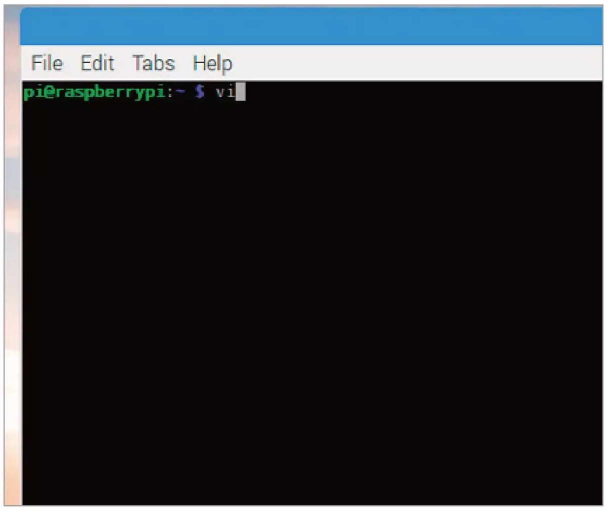
You will come across a lot of text files in Linux and to be able to edit or create a text file, you need a good text editor. Linux has many but here are some in action on the Raspberry Pi.

STEP 1 The first text editor for the Raspberry Pi is the default desktop environment app: Leafpad. To use, you can either double-click an existing text file or click the Raspberry Pi menu icon (in the top left of the desktop) and from the Accessories menu, choose Text Editor.



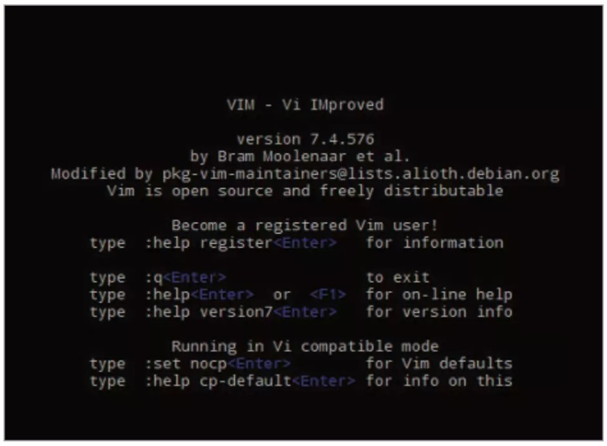
```
I remember
The dark woods, masking slopes of sombre hills
The grey clouds' leaden everlasting arch;
The dusky streams that flowed without a sound,
And the lone winds that whispered down the pas

Vista on vista marching, hills on hills,
Slope beyond slope, each dark with sullen tree
Our gaunt land lay. So when a man climbed up
A rugged peak and gazed, his shaded eye
Saw but the endless vista - hill on hill,
Slope beyond slope, each hooded like its broth

It was a gloomy land that seemed to hold
All winds and clouds and dreams that shun the
With bare boughs rattling in the lonesome wind
And the dark woodlands brooding over all,
Not even lightened by the rare dim sun
Which made squat shadows out of men; they call
Cimmeria, land of Darkness and deep Night.

It was so long ago and far away
I have forgot the very name men called me.
The axe and flint-tipped spear are like a drea
And hunts and wars are shadows. I recall
Only the stillness of that sombre land;
The clouds that piled forever on the hills,
The dimness of the everlasting woods.
Cimmeria, land of Darkness and the Night.

Oh, soul of mine, born out of shadowed hills,
To clouds and winds and ghosts that shun the s
How many deaths shall serve to break at last
This heritage which wraps me in the grey
Apparel of ghosts? I search my heart and find
Cimmeria, land of Darkness and the Night
```

STEP 2 From the Terminal there are even more options, although using the correct command, you can launch any of the desktop apps via the Terminal. One of the simplest, and a classic text editor that's carried over from the Unix days, is vi. In the Terminal, enter: `vi`.



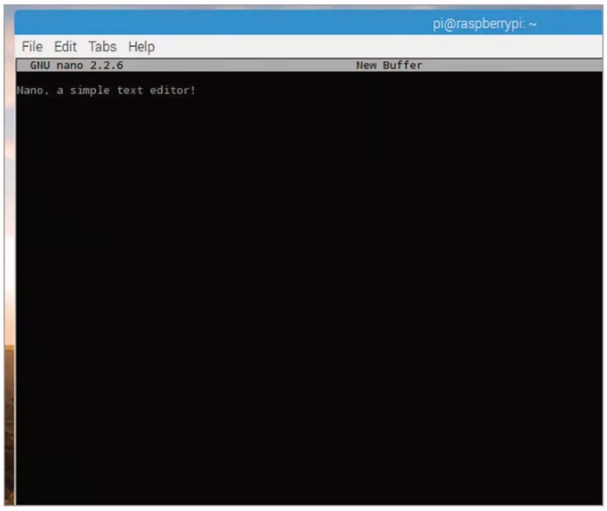
```
pi@raspberrypi:~ $ vi
```

STEP 3 Vi is the original Unix command but in this case it launches VIM, the new Linux version of Vi. Although simple looking, Vi is considered, even by today's standards, to be one of the most widely used text editors, There's a lot you can do with it, so check out the man pages for more Vi information.



```
                 VIM - Vi IMproved

                  version 7.4.576
              by Bram Moolenaar et al.
   Modified by pkg-vim-maintainers@lists.alioth.debian.org
        Vim is open source and freely distributable

               Become a registered Vim user!
    type  :help register<Enter>   for information

    type  :q<Enter>               to exit
    type  :help<Enter>  or  <F1>  for on-line help
    type  :help version7<Enter>   for version info

               Running in Vi compatible mode
    type  :set nocp<Enter>        for Vim defaults
    type  :help cp-default<Enter> for info on this
```

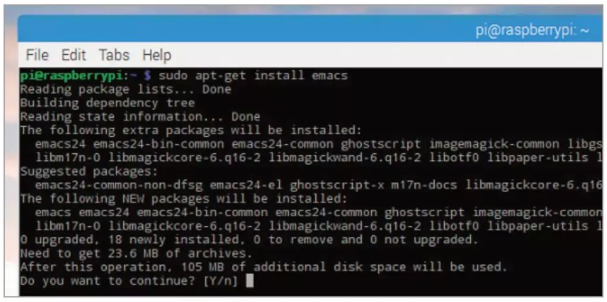
STEP 4 Nano is another favourite, and simple, text editor available for Linux. Enter: `nano` into the Terminal to launch it. You can use Nano for editing code, creating scripts or writing your own help files. To exit Nano, press Ctrl + X, followed by Y to save the file or N to exit without saving.



```
  GNU nano 2.2.6                        New Buffer

Nano, a simple text editor!
```

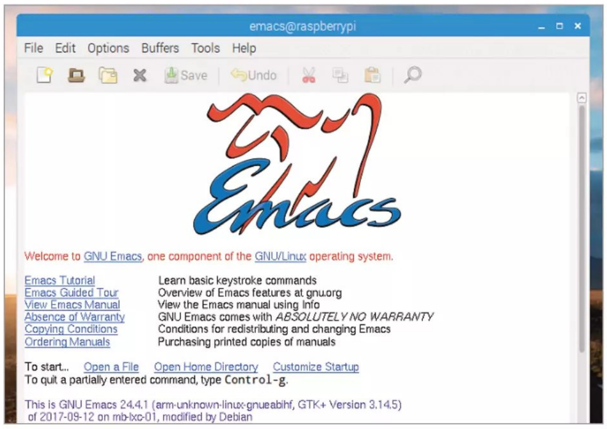
STEP 5 Emacs, or GNU Emacs, is an extensible and customisable, self-documenting, real-time display editor. It's a fantastic text editor and one that's worth getting used to as soon as you can. Sadly, it's not installed on the Pi by default, so you'll need to install it. In the Terminal, enter:

```
sudo apt-get install emacs
```



```
pi@raspberrypi:~ $ sudo apt-get install emacs
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  emacs24 emacs24-bin-common emacs24-common ghostscript imagemagick-common libgs
  libm17n-0 libmagickcore-6.q16-2 libmagickwand-6.q16-2 libotf0 libpaper-utils l
Suggested packages:
  emacs24-common-non-dfsg emacs24-el ghostscript-x m17n-docs libmagickcore-6.q16
The following NEW packages will be installed:
  emacs emacs24 emacs24-bin-common emacs24-common ghostscript imagemagick-common
  libm17n-0 libmagickcore-6.q16-2 libmagickwand-6.q16-2 libotf0 libpaper-utils l
0 upgraded, 18 newly installed, 0 to remove and 0 not upgraded.
Need to get 23.6 MB of archives.
After this operation, 105 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

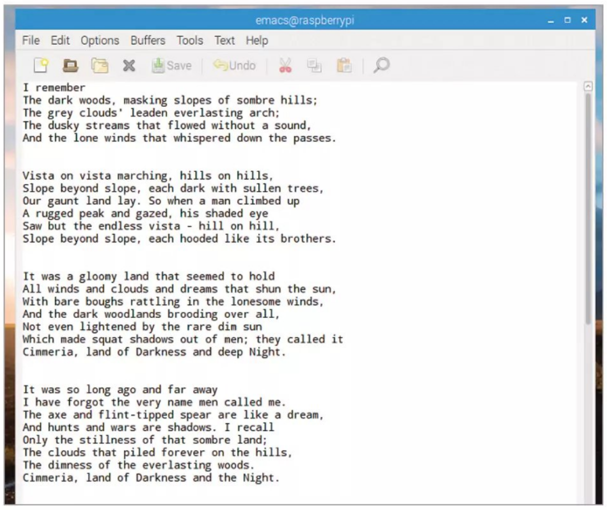
STEP 6 The previous command contacts the Debian (Raspbian is based on a Debian Linux distribution) repositories and pulls down the information needed to install Emacs. When the Pi asks to continue with the installation, press Y. This installs the latest version and when it's done, you'll be back to the command prompt.



Welcome to GNU Emacs, one component of the GNU/Linux operating system.

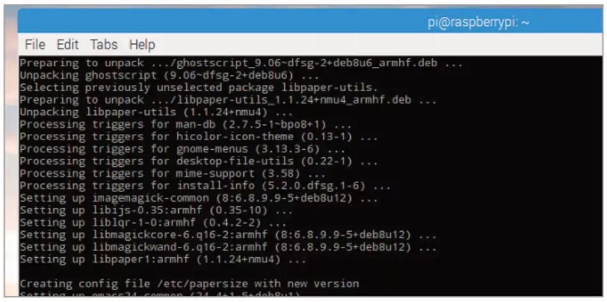
```
Emacs Tutorial          Learn basic keystroke commands
Emacs Guided Tour       Overview of Emacs features at gnu.org
View Emacs Manual       View the Emacs manual using Info
Absence of Warranty     GNU Emacs comes with ABSOLUTELY NO WARRANTY
Copying Conditions      Conditions for redistributing and changing Emacs
Ordering Manuals        Purchasing printed copies of manuals

To start...   Open a File   Open Home Directory   Customize Startup
To quit a partially entered command, type Control-g.

This is GNU Emacs 24.4.1 (arm-unknown-linux-gnueabihf, GTK+ Version 3.14.5)
 of 2017-09-12 on mb-lxc-01, modified by Debian
```

STEP 7 Once the installation is complete, enter: `emacs` into the Terminal. The Emacs splash screen opens in a new window, offering a tutorial (which we recommend you run through) and a guided tour amongst other information.



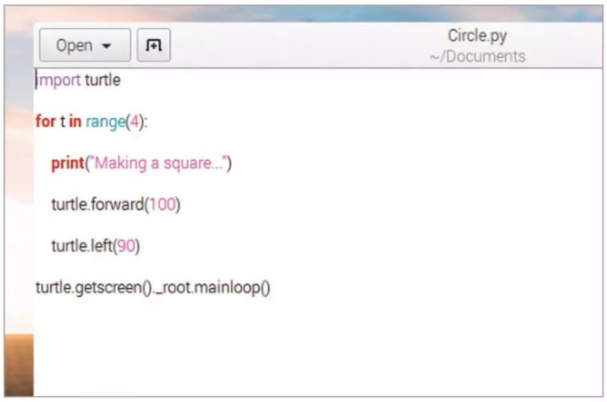
```
I remember
The dark woods, masking slopes of sombre hills;
The grey clouds' leaden everlasting arch;
The dusky streams that flowed without a sound,
And the lone winds that whispered down the passes.


Vista on vista marching, hills on hills,
Slope beyond slope, each dark with sullen trees,
Our gaunt land lay. So when a man climbed up
A rugged peak and gazed, his shaded eye
Saw but the endless vista - hill on hill,
Slope beyond slope, each hooded like its brothers.


It was a gloomy land that seemed to hold
All winds and clouds and dreams that shun the sun,
With bare boughs rattling in the lonesome winds,
And the dark woodlands brooding over all,
Not even lightened by the rare dim sun
Which made squat shadows out of men; they called it
Cimmeria, land of Darkness and deep Night.


It was so long ago and far away
I have forgot the very name men called me.
The axe and flint-tipped spear are like a dream,
And hunts and wars are shadows. I recall
Only the stillness of that sombre land;
The clouds that piled forever on the hills,
The dimness of the everlasting woods.
Cimmeria, land of Darkness and the Night.
```

STEP 8 Emacs can offer an uncomplicated view of your text file or one with a plethora of information regarding the structure of the file in question; it's up to you to work out your own preference. There's also a hidden text adventure in Emacs, which we cover later in this book, why not see if you can find it without our help.



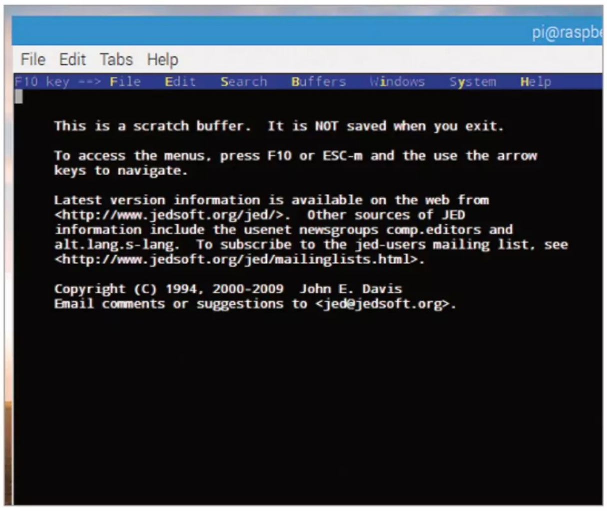
```
Preparing to unpack .../ghostscript_9.06~dfsg-2+deb8u6_armhf.deb ...
Unpacking ghostscript (9.06~dfsg-2+deb8u6) ...
Selecting previously unselected package libpaper-utils.
Preparing to unpack .../libpaper-utils_1.1.24+nmu4_armhf.deb ...
Unpacking libpaper-utils (1.1.24+nmu4) ...
Processing triggers for man-db (2.7.5-1~bpo8+1) ...
Processing triggers for hicolor-icon-theme (0.13-1) ...
Processing triggers for gnome-menus (3.13.3-6) ...
Processing triggers for desktop-file-utils (0.22-1) ...
Processing triggers for mime-support (3.58) ...
Processing triggers for install-info (5.2.0.dfsg.1-6) ...
Setting up imagemagick-common (8:6.8.9.9-5+deb8u12) ...
Setting up libijs-0.35:armhf (0.35-10) ...
Setting up liblqr-1-0:armhf (0.4.2-2) ...
Setting up libmagickcore-6.q16-2:armhf (8:6.8.9.9-5+deb8u12) ...
Setting up libmagickwand-6.q16-2:armhf (8:6.8.9.9-5+deb8u12) ...
Setting up libpaper1:armhf (1.1.24+nmu4) ...

Creating config file /etc/papersize with new version
```

STEP 9 Gedit is another excellent text editor for Linux. Again, it's not installed by default on the Raspberry Pi; however, by entering: `sudo apt-get install gedit` and accepting the installation, the program can be on the Pi in a matter of seconds. Once it's installed, use `gedit` in the Terminal to launch it. Gedit is a great text editor for coding.



```
import turtle

for t in range(4):

  print("Making a square...")

  turtle.forward(100)

  turtle.left(90)

turtle.getscreen()._root.mainloop()
```

STEP 10 Finally, Jed is an Emacs-like, cross-platform text editor that's lightweight and comes with a wealth of features. To install it, enter: `sudo apt-get install jed`. Accept the installation and when it's complete, use: `jed` to launch.



```
F10 key ==> File   Edit   Search   Buffers   Windows   System   Help

     This is a scratch buffer.  It is NOT saved when you exit.

     To access the menus, press F10 or ESC-m and the use the arrow
     keys to navigate.

     Latest version information is available on the web from
     <http://www.jedsoft.org/jed/>.  Other sources of JED
     information include the usenet newsgroups comp.editors and
     alt.lang.s-lang.  To subscribe to the jed-users mailing list, see
     <http://www.jedsoft.org/jed/mailinglists.html>.

     Copyright (C) 1994, 2000-2009  John E. Davis
     Email comments or suggestions to <jed@jedsoft.org>.
```

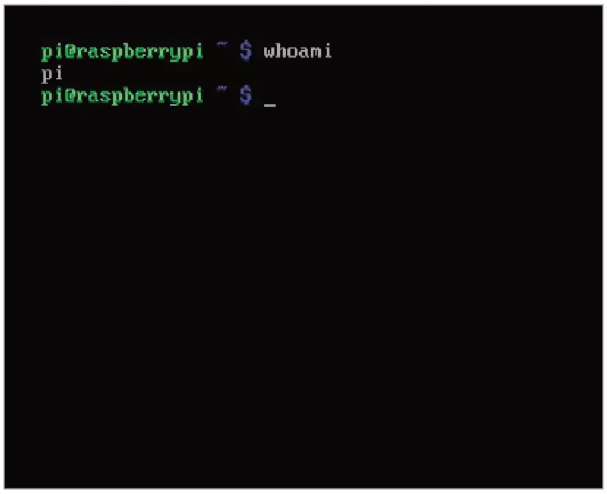
Getting to Know Users

You might think you're the only person using your Raspberry Pi but there are several different users and even groups of users. Your main account is normally called Pi and there is an account above it called root, which is more powerful. You can also create users and groups.

WHAT IS A USER?

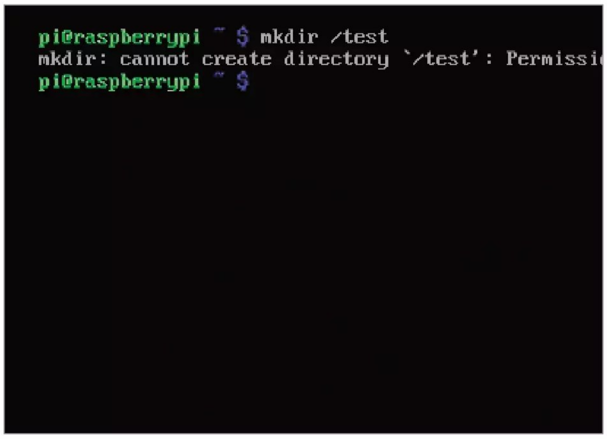
An important part of using Linux is the concept of users and understanding which user you are and which group you belong to. Like all modern computers, you can have multiple user accounts with each having different levels of access.

STEP 1 The first thing you need to do is get a concept of which user you are. Enter: `whoami` into the command line and press return. It should say "pi" (unless you set up your account name differently during setup). The "whoami" command might seem a bit simplistic, but it comes in very handy sometimes.



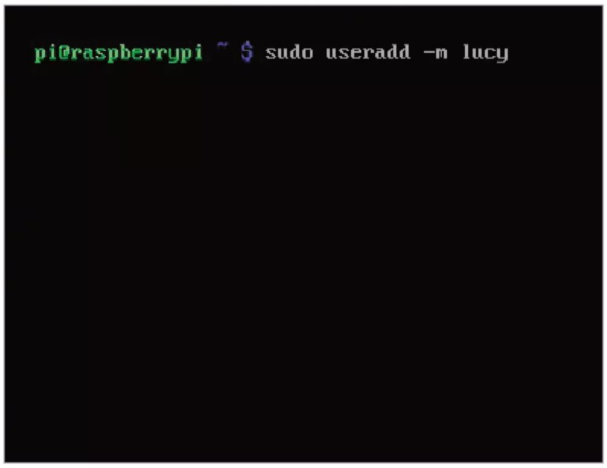
```
pi@raspberrypi ~ $ whoami
pi
pi@raspberrypi ~ $ _
```

STEP 2 When you are working in Linux, from time to time a 'Permission denied' error will occur, typically when you try to create, edit or execute (run) a file or directory outside of your area of privilege. If you want to see this, enter: `mkdir /testdir`. Attempting to create a new directory in your root directory isn't allowed.



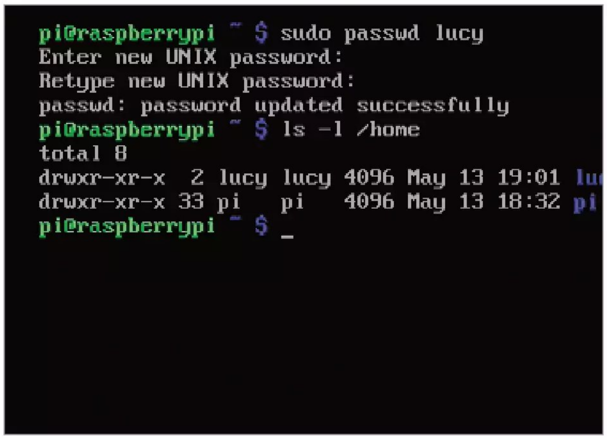
```
pi@raspberrypi ~ $ mkdir /test
mkdir: cannot create directory `/test': Permissi
pi@raspberrypi ~ $
```

STEP 3 To allow this, you need to use the sudo command. Sudo loosely stands for Substitute User Do; essentially it's the highest level of access to the system and you've already installed text editors using sudo. You'll come across sudo frequently in Linux, so let's create a second account to get the hang of it. Enter: `sudo useradd -m lucy` (or pick your name).



```
pi@raspberrypi ~ $ sudo useradd -m lucy
```

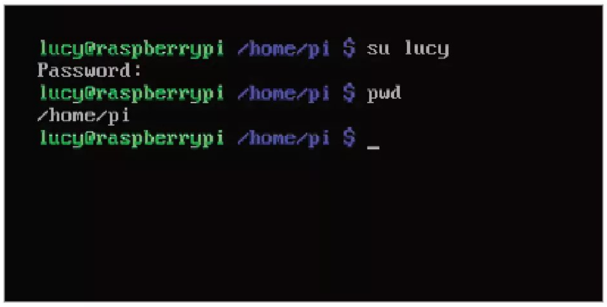
STEP 4 Now add a password for the new account. Enter: `sudo passwd lucy` and enter: a short password. Retype the same password and you'll now have two accounts on your Raspberry Pi. Now enter: `ls -l /home` to view the home directories for both users. Notice that the lucy directory lists lucy as the owner and group; and pi directory is belongs to pi.



```
pi@raspberrypi ~ $ sudo passwd lucy
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
pi@raspberrypi ~ $ ls -l /home
total 8
drwxr-xr-x  2 lucy lucy 4096 May 13 19:01 lu
drwxr-xr-x 33 pi   pi   4096 May 13 18:32 pi
pi@raspberrypi ~ $ _
```

STEP 5

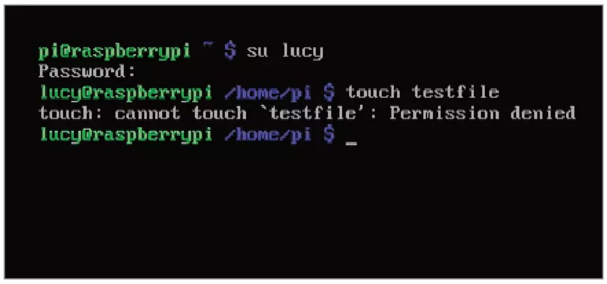
Let's try switching to our new account. Enter: **`su lucy`** and enter the password you just created for that account. Notice that the command line now says "lucy@raspberrypi" but the working directory is "still /home/pi" (check this using "pwd"). Enter: **`whoami`** to confirm that you are now the new user.



```
lucy@raspberrypi /home/pi $ su lucy
Password:
lucy@raspberrypi /home/pi $ pwd
/home/pi
lucy@raspberrypi /home/pi $ _
```

STEP 6

We'll look at permissions in the next tutorial, but for now try to create a file as before. Enter: **`touch testfile`** to create a file. It will say "touch: cannot touch 'testfile': Permission denied". This is because your new user account doesn't have the right to create files in the /home/pi directory. Enter: **`su pi`** to switch back to your pi account.



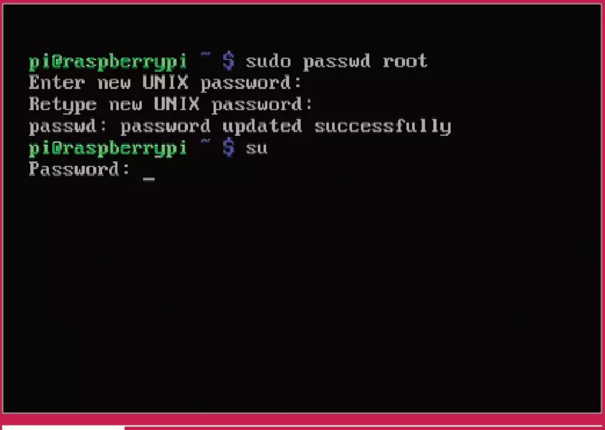
```
pi@raspberrypi ~ $ su lucy
Password:
lucy@raspberrypi /home/pi $ touch testfile
touch: cannot touch `testfile': Permission denied
lucy@raspberrypi /home/pi $ _
```

GETTING SUDO

We now have two accounts on our Raspberry Pi: lucy and pi. The lucy account can edit files in /home/lucy and the pi account can edit files in /home/pi. But there's also a third account, called "root", that sits above both lucy and pi. It can edit files anywhere.

STEP 1

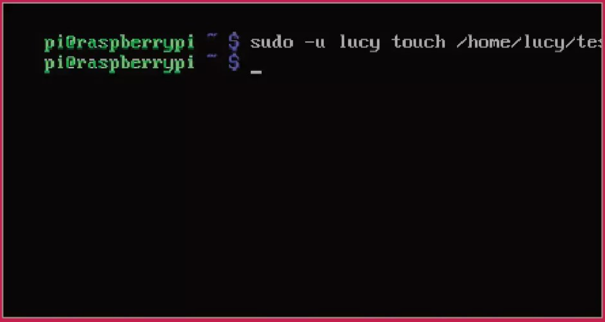
The root account is all-powerful. It is possible, but not recommended, to switch to the root account, although you'll need to give it a password first (using "sudo passwd root"). Then just type "su" to switch to root. Please don't do this though: knowledge is a good thing but it's safer and wiser to use sudo instead.



```
pi@raspberrypi ~ $ sudo passwd root
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
pi@raspberrypi ~ $ su
Password: _
```

STEP 2

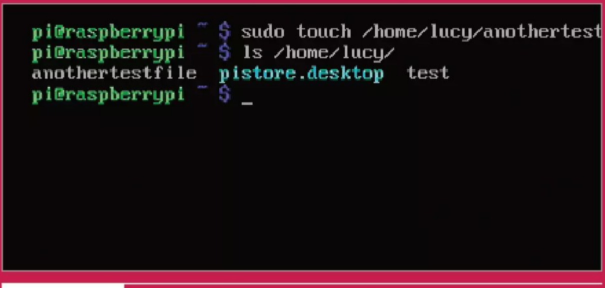
Most people think sudo stands for "super user", but it stands for "substitute user do". It enables you to perform a command as another user. Enter: **`sudo -u lucy touch /home/lucy/test`** to create a file inside the lucy home directory. You won't get an error because the lucy user has permission to edit that directory.



```
pi@raspberrypi ~ $ sudo -u lucy touch /home/lucy/te
pi@raspberrypi ~ $ _
```

STEP 3

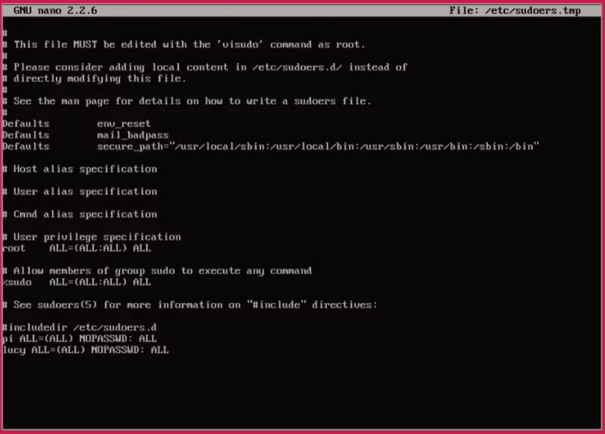
It's rare that you use sudo to substitute another user. If you don't specify a user using the "-"u option with a username it defaults to the root account, as if you'd typed "sudo -u root". Enter: **`sudo touch /home/lucy/anothertestfile`** to create a file in the lucy directory while still using the pi account.



```
pi@raspberrypi ~ $ sudo touch /home/lucy/anothertest
pi@raspberrypi ~ $ ls /home/lucy/
anothertestfile  pistore.desktop  test
pi@raspberrypi ~ $ _
```

STEP 4

This step is optional. Only the pi user can use sudo. If we want to give the lucy account sudo privileges, it needs to be added to the sudoers file. Enter: **`sudo visudo`** to view the sudoers file. Add **`lucy ALL=(ALL) NOPASSWD: ALL`** to the last line and use Control+O to output the file. Remove the ".tmp" that is added to the file name as a security measure. Note that most accounts are not added to the sudoers file as a matter of course.



```
  GNU nano 2.2.6                                              File: /etc/sudoers.tmp

#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d
pi ALL=(ALL) NOPASSWD: ALL
lucy ALL=(ALL) NOPASSWD: ALL
```

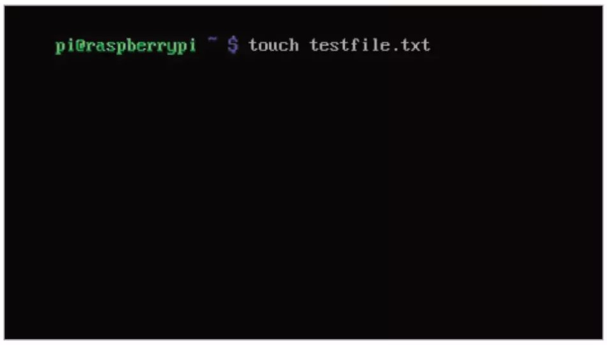
Ownership and Permissions

Once you've got the hang of users, you need to learn about ownership and permissions. Different users have different areas of ownership and can do different things with each file. Permissions in Linux can be quite complex but with careful thought it's not too difficult.

OWNER AND PRIVILEGE

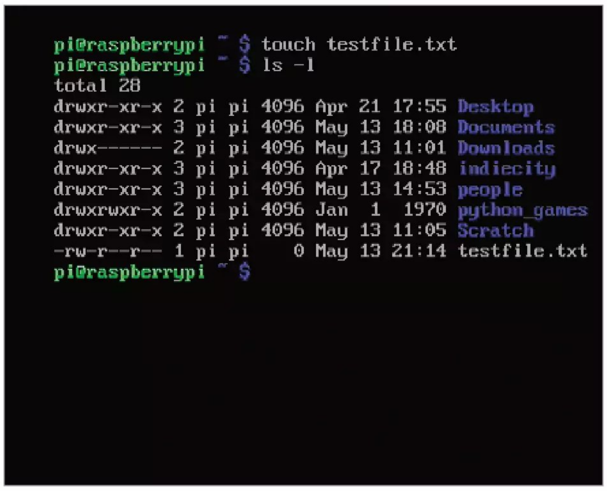
Each user account in Linux is an owner of a section of the filesystem: their Home area. Within this area, they do what they like (within reason), as they have owner privileges. Elsewhere though, they usually just have read-only privileges.

STEP 1 If you followed the previous tutorial you should now have two accounts on your Raspberry Pi. One called "pi" and the other with a name (Lucy in our case). An essential aspect of Linux is the idea of file and directory ownership; who owns, and has access, to what. You need a test file so enter: `touch testfile`.



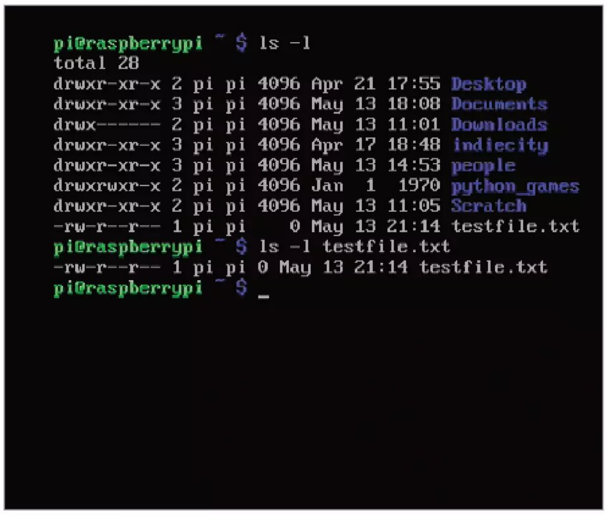
```
pi@raspberrypi ~ $ touch testfile.txt
```

STEP 2 Now enter: `ls -l` and let's have a good look at the default permissions file. Our testfile.txt files starts with the text "-rw-r--r—". Start with the first letter, which is a dash '-'. All the other items in our home directory are directories. You can tell because they are blue, and our testfile.txt file is white.



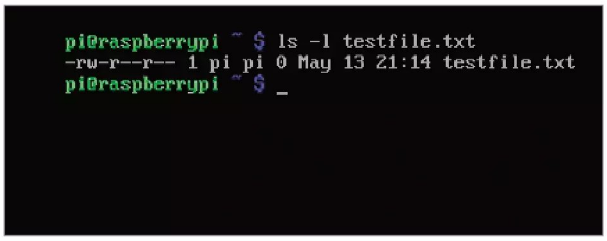
```
pi@raspberrypi ~ $ touch testfile.txt
pi@raspberrypi ~ $ ls -l
total 28
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 3 pi pi 4096 May 13 18:08 Documents
drwx------ 2 pi pi 4096 May 13 11:01 Downloads
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
drwxr-xr-x 3 pi pi 4096 May 13 14:53 people
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 May 13 11:05 Scratch
-rw-r--r-- 1 pi pi    0 May 13 21:14 testfile.txt
pi@raspberrypi ~ $
```

STEP 3 The first letter in the permissions also indicates a directory or file. Notice that all the other files start with a 'd' and our textfile.txt file starts with a '-'. That's what the first letter means. It's either a 'd', in which case it's a directory, or a '-', in which case it's not; it's a file. Enter: `ls -l testfile.txt` to view the permissions for just this file.



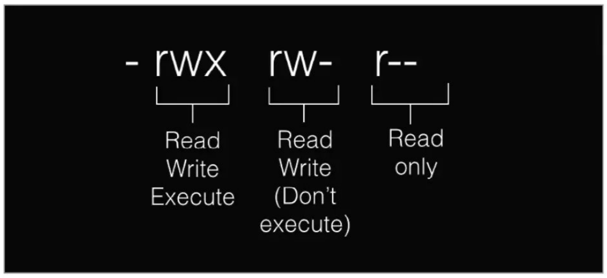
```
pi@raspberrypi ~ $ ls -l
total 28
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 3 pi pi 4096 May 13 18:08 Documents
drwx------ 2 pi pi 4096 May 13 11:01 Downloads
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
drwxr-xr-x 3 pi pi 4096 May 13 14:53 people
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 May 13 11:05 Scratch
-rw-r--r-- 1 pi pi    0 May 13 21:14 testfile.txt
pi@raspberrypi ~ $ ls -l testfile.txt
-rw-r--r-- 1 pi pi 0 May 13 21:14 testfile.txt
pi@raspberrypi ~ $ _
```

STEP 4 The next nine letters of the permissions are known as "alpha notation" because they let you know the permissions using letters. Each permission is either on, in which case you see a letter, or it is off, in which case you see a dash. The letter doesn't change for each place. So the first permission - the second letter after the directory one - is either an 'r' or a '-'. It's never any other letter.

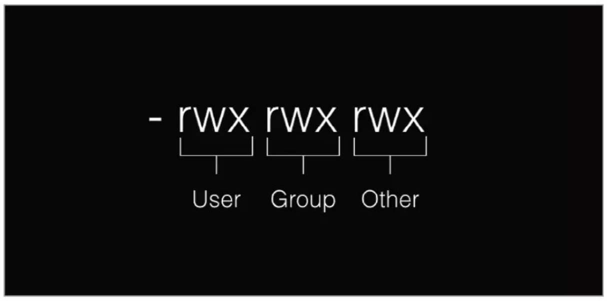


```
pi@raspberrypi ~ $ ls -l testfile.txt
-rw-r--r-- 1 pi pi 0 May 13 21:14 testfile.txt
pi@raspberrypi ~ $ _
```

STEP 5 The 'r' means that particular permission is read and it's set to On. The nine letters here are divided into three groups of three letters: r, w, x. They stand for read, write and execute (run). Read means the file can be viewed (using cat or nano); w means the file can be edited or moved, and x means the file - typically a script or program - can be run.



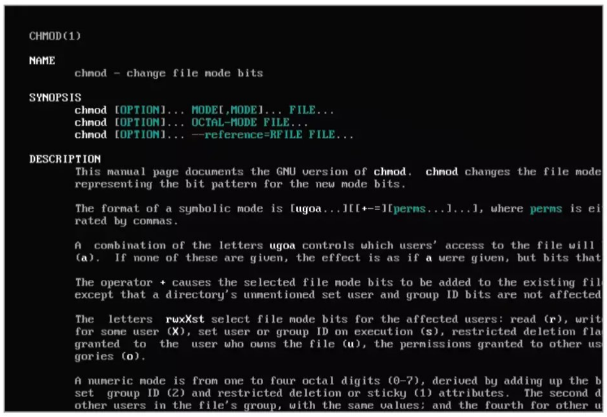
STEP 6 The presence of r, w, or x means that this aspect is possible, a dash means it isn't. Our testfile.txt has no x letter; so if it were a script it wouldn't run. So why are there so many letters? Why not just three; read, write and execute? The three blocks of three letters are for different sets of people: user, group and other.



CHANGING PERMISSIONS

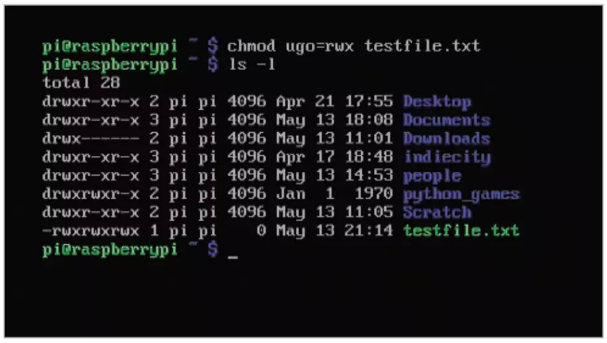
Now that you know how groups of permissions work, it's time to look at how to change them.

STEP 1 The first block of three is the most important. This is the user who owns the file (typically pi); the second is for other people in the same group as the user, and the third is for other people on the system. Permissions are changed using the chmod (change file mode bit) command. Enter: `man chmod` to look at the manual.



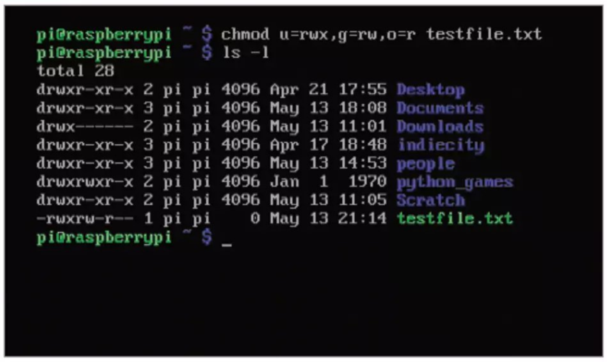
```
CHMOD(1)

NAME
       chmod - change file mode bits

SYNOPSIS
       chmod [OPTION]... MODE[,MODE]... FILE...
       chmod [OPTION]... OCTAL-MODE FILE...
       chmod [OPTION]... --reference=RFILE FILE...

DESCRIPTION
       This manual page documents the GNU version of chmod.  chmod changes the file mode
       representing the bit pattern for the new mode bits.

       The format of a symbolic mode is [ugoa...][[+-=][perms...]...], where perms is ei
       rated by commas.

       A  combination of the letters ugoa controls which users' access to the file will
       (a).  If none of these are given, the effect is as if a were given, but bits that

       The operator + causes the selected file mode bits to be added to the existing fil
       except that a directory's unmentioned set user and group ID bits are not affected

       The  letters  rwxXst select file mode bits for the affected users: read (r), writ
       for some user (X), set user or group ID on execution (s), restricted deletion fla
       granted  to  the  user who owns the file (u), the permissions granted to other us
       gories (o).

       A numeric mode is from one to four octal digits (0-7), derived by adding up the b
       set  group ID (2) and restricted deletion or sticky (1) attributes.  The second d
       other users in the file's group, with the same values; and the fourth for other u
```

STEP 2 The chmod command is one of the trickier ones to understand. There are two ways you can adjust permissions; the first is using chmod with an option to target one of the three groups: owner, group, other. For these you use u, g or o followed by = and the letters or dashes you want. So enter: `chmod ugo=rx testfile.txt` to make all three groups read, write and execute.



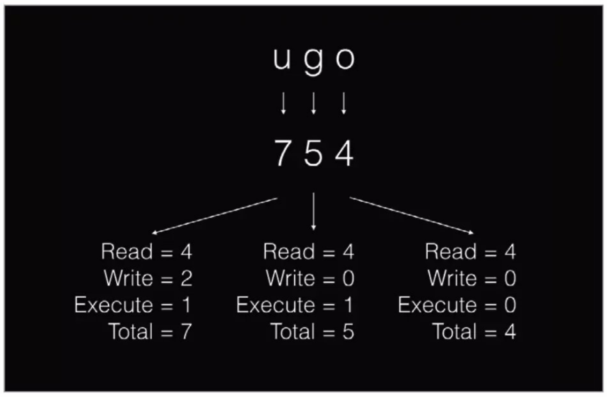
```
pi@raspberrypi ~ $ chmod ugo=rwx testfile.txt
pi@raspberrypi ~ $ ls -l
total 28
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 3 pi pi 4096 May 13 18:08 Documents
drwx------ 2 pi pi 4096 May 13 11:01 Downloads
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
drwxr-xr-x 3 pi pi 4096 May 13 14:53 people
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 May 13 11:05 Scratch
-rwxrwxrwx 1 pi pi    0 May 13 21:14 testfile.txt
pi@raspberrypi ~ $ _
```

STEP 3 Turning everything on is probably overkill, so you need to target each group. Do this by putting commas between each mode option. Enter: `chmod u=rwx,g=rw,o=r testfile.txt` to give users read, write and execute privileges, user read and write and other just read. Enter: `ls -l` to see your handiwork.



```
pi@raspberrypi ~ $ chmod u=rwx,g=rw,o=r testfile.txt
pi@raspberrypi ~ $ ls -l
total 28
drwxr-xr-x 2 pi pi 4096 Apr 21 17:55 Desktop
drwxr-xr-x 3 pi pi 4096 May 13 18:08 Documents
drwx------ 2 pi pi 4096 May 13 11:01 Downloads
drwxr-xr-x 3 pi pi 4096 Apr 17 18:48 indiecity
drwxr-xr-x 3 pi pi 4096 May 13 14:53 people
drwxrwxr-x 2 pi pi 4096 Jan  1  1970 python_games
drwxr-xr-x 2 pi pi 4096 May 13 11:05 Scratch
-rwxrw-r-- 1 pi pi    0 May 13 21:14 testfile.txt
pi@raspberrypi ~ $ _
```

STEP 4 Alpha notation is fine, but many Linux admins use octal notation instead. This is a three-digit number that represents permissions. This is the formula: read=4, write=2 and execute=1 and you add them up for each group, therefore if a group is read, write and execute it's 7, if it's read and write it's 6 or if it's just execute it's 1. A popular option is 755. Enter: `chmod 755 testfile.txt` to change the file using octal notation.



Useful System and Disk Commands

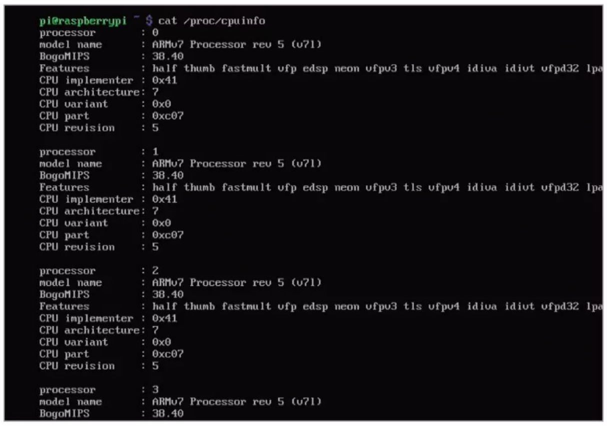
Understanding these core Linux commands will enable you to not only master the inner workings of your Raspberry Pi but also to transfer those skills to other Linux distros, such as Ubuntu or Linux Mint.

LOTS OF LINUX

Linux is a huge and versatile command line language and there are hundreds of commands you can learn and use. Here are a few that can help you get more from your Raspberry Pi.

STEP 1

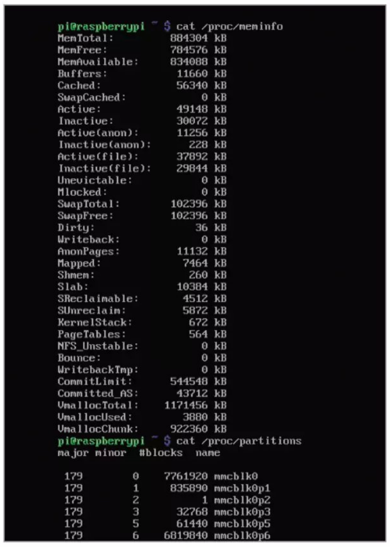
The Raspberry Pi is a great little computer, so let's start by getting some information. Enter: `cat /proc/cpuinfo` to view some details on your Raspberry Pi processors. If you have a Raspberry Pi 3 you will see four processors, along with the model name and other info.



```
pi@raspberrypi ~ $ cat /proc/cpuinfo
processor       : 0
model name      : ARMv7 Processor rev 5 (v7l)
BogoMIPS        : 38.40
Features        : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd32 lpa
CPU implementer : 0x41
CPU architecture: 7
CPU variant     : 0x0
CPU part        : 0xc07
CPU revision    : 5

processor       : 1
model name      : ARMv7 Processor rev 5 (v7l)
BogoMIPS        : 38.40
Features        : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd32 lpa
CPU implementer : 0x41
CPU architecture: 7
CPU variant     : 0x0
CPU part        : 0xc07
CPU revision    : 5

processor       : 2
model name      : ARMv7 Processor rev 5 (v7l)
BogoMIPS        : 38.40
Features        : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd32 lpa
CPU implementer : 0x41
CPU architecture: 7
CPU variant     : 0x0
CPU part        : 0xc07
CPU revision    : 5

processor       : 3
model name      : ARMv7 Processor rev 5 (v7l)
BogoMIPS        : 38.40
```

STEP 2

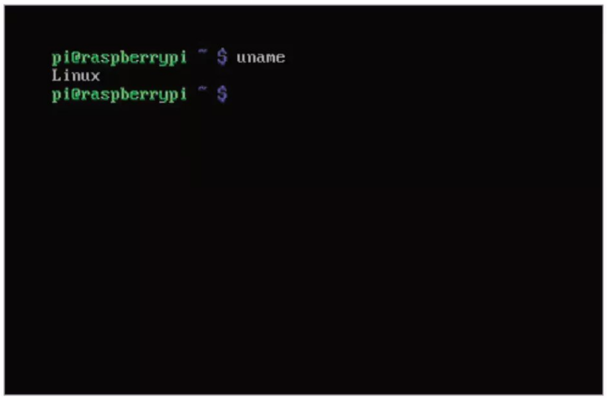
Remember that cat is used to list the contents of a text file, which is what cpuinfo is. There are other text files with system info available. Try "cat /proc/meminfo" to get information about your memory, "cat /proc/partitions" for information about your SD card, and "cat /proc/version" shows which version of Raspberry Pi you are using.



```
pi@raspberrypi ~ $ cat /proc/meminfo
MemTotal:         884304 kB
MemFree:          784576 kB
MemAvailable:     834088 kB
Buffers:           11660 kB
Cached:            56340 kB
SwapCached:            0 kB
Active:            49148 kB
Inactive:          30072 kB
Active(anon):      11256 kB
Inactive(anon):      228 kB
Active(file):      37892 kB
Inactive(file):    29844 kB
Unevictable:           0 kB
Mlocked:               0 kB
SwapTotal:        102396 kB
SwapFree:         102396 kB
Dirty:                36 kB
Writeback:             0 kB
AnonPages:         11132 kB
Mapped:             7464 kB
Shmem:               260 kB
Slab:              10384 kB
SReclaimable:       4512 kB
SUnreclaim:         5872 kB
KernelStack:         672 kB
PageTables:          564 kB
NFS_Unstable:          0 kB
Bounce:                0 kB
WritebackTmp:          0 kB
CommitLimit:      544548 kB
Committed_AS:      43712 kB
VmallocTotal:    1171456 kB
VmallocUsed:        3880 kB
VmallocChunk:     922360 kB
pi@raspberrypi ~ $ cat /proc/partitions
major minor  #blocks  name

 179        0    7761920 mmcblk0
 179        1     835890 mmcblk0p1
 179        2          1 mmcblk0p2
 179        3      32768 mmcblk0p3
 179        5      61440 mmcblk0p5
 179        6    6819840 mmcblk0p6
```

STEP 3

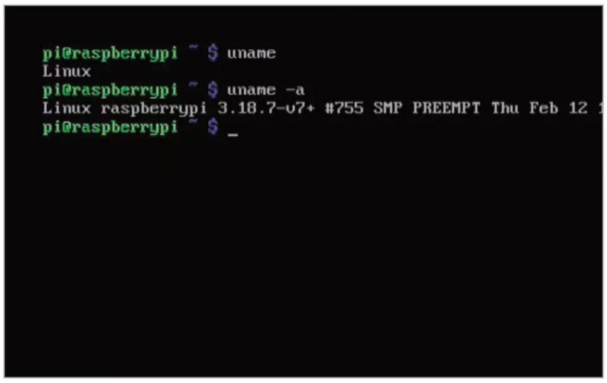
Enter: `uname` to view the name of the operating system's kernel, this is the element that sits between the interface and hardware. Just as you would suspect, the response from the command is Linux, as Raspbian is a Linux distro, which in itself is based on another Linux distro called Debian. While it may sound complicated, it actually demonstrates how versatile Linux is.



```
pi@raspberrypi ~ $ uname
Linux
pi@raspberrypi ~ $
```

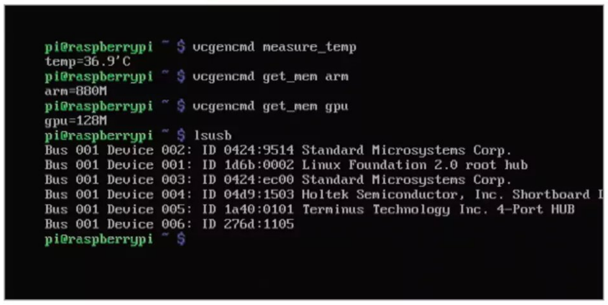
STEP 4

Enter: `uname -a` to view some more detailed information. Here you'll see the kernel name, hostname and kernel version (3.18.7-v7 on ours). If you have a Raspberry Pi 2 you'll see SMP (symmetric multiprocessing), followed by the system date, CPU architecture and operating system (GNU/Linux).



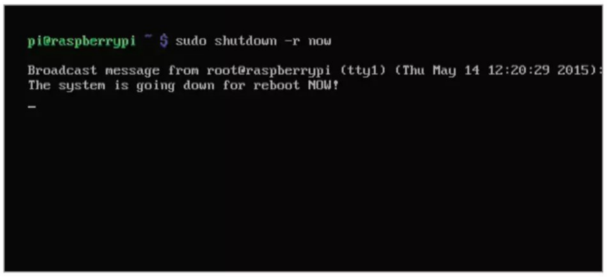
```
pi@raspberrypi ~ $ uname
Linux
pi@raspberrypi ~ $ uname -a
Linux raspberrypi 3.18.7-v7+ #755 SMP PREEMPT Thu Feb 12 1
pi@raspberrypi ~ $ _
```

STEP 5 Enter: `vcgencmd measure_temp` to view the current operating system temperature of your Raspberry Pi. Enter: `vcgencmd get_mem arm` to view the RAM available, and `vcgencmd get_mem gpu` to view the memory available to the graphics chip. Finally try `ls usb` to view a list of attached USB devices.



```
pi@raspberrypi ~ $ vcgencmd measure_temp
temp=36.9'C
pi@raspberrypi ~ $ vcgencmd get_mem arm
arm=880M
pi@raspberrypi ~ $ vcgencmd get_mem gpu
gpu=128M
pi@raspberrypi ~ $ lsusb
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp.
Bus 001 Device 004: ID 04d9:1503 Holtek Semiconductor, Inc. Shortboard I
Bus 001 Device 005: ID 1a40:0101 Terminus Technology Inc. 4-Port HUB
Bus 001 Device 006: ID 276d:1105
pi@raspberrypi ~ $
```

STEP 6 One command you might be wondering about is how to switch off or restart your Raspberry Pi from the command line. Don't just hit the power switch. Enter: `sudo showdown -h` now to shut down the Raspberry Pi (the "-h" option stands for "halt"), or enter: `sudo shutdown -r` now to restart your Raspberry Pi.

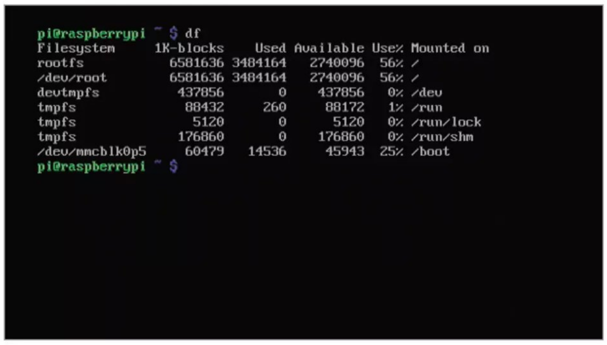


```
pi@raspberrypi ~ $ sudo shutdown -r now

Broadcast message from root@raspberrypi (tty1) (Thu May 14 12:20:29 2015):
The system is going down for reboot NOW!

_
```

DISK COMMANDS

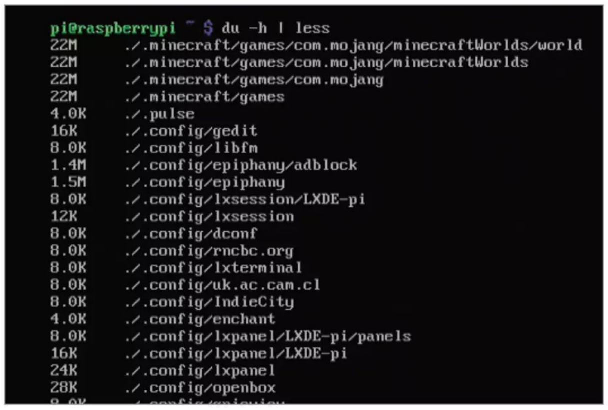
Learn the two commands that enable you to view your disk space and the files on it: df (disk free space) and du (disk usage). With these two commands you can view the file usage on your SD card.

STEP 1 Start by entering: `df` in the command line. It returns a list of the volumes contained on your SD card. You might be wondering what a volume is. It's best to think of your SD card as the drive. This contains partitions, which is where you split one drive to act like two or more drives. And each partition can contain volumes, which are storage spaces.



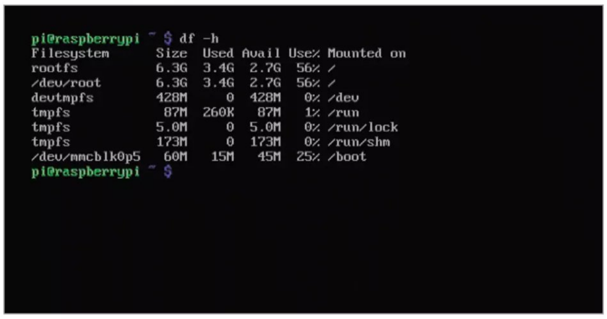
```
pi@raspberrypi ~ $ df
Filesystem     1K-blocks    Used Available Use% Mounted on
rootfs           6581636 3484164   2740096  56% /
/dev/root        6581636 3484164   2740096  56% /
devtmpfs          437856       0    437856   0% /dev
tmpfs              88432     260     88172   1% /run
tmpfs               5120       0      5120   0% /run/lock
tmpfs             176860       0    176860   0% /run/shm
/dev/mmcblk0p5     60479   14536     45943  25% /boot
pi@raspberrypi ~ $
```

STEP 2 Enter: `df -h` to get the list in human readable form. The first two lines should read "rootfs" and "/dev/root" and have matching Size, Used, Avail and Use% listings. This is the main drive, and is an indication of how much space you have used, and have free, on your Raspbian OS. The other volumes are for booting and initialising devices (you can ignore these for now).



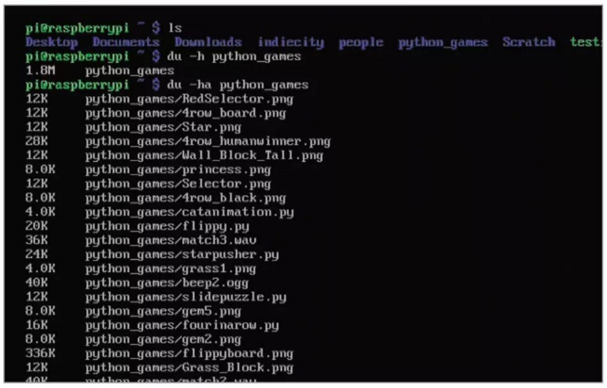
```
pi@raspberrypi ~ $ du -h | less
22M     ./.minecraft/games/com.mojang/minecraftWorlds/world
22M     ./.minecraft/games/com.mojang/minecraftWorlds
22M     ./.minecraft/games/com.mojang
22M     ./.minecraft/games
4.0K    ./.pulse
16K     ./.config/gedit
8.0K    ./.config/libfm
1.4M    ./.config/epiphany/adblock
1.5M    ./.config/epiphany
8.0K    ./.config/lxsession/LXDE-pi
12K     ./.config/lxsession
8.0K    ./.config/dconf
8.0K    ./.config/rncbc.org
8.0K    ./.config/lxterminal
8.0K    ./.config/uk.ac.cam.cl
8.0K    ./.config/IndieCity
4.0K    ./.config/enchant
8.0K    ./.config/lxpanel/LXDE-pi/panels
16K     ./.config/lxpanel/LXDE-pi
24K     ./.config/lxpanel
28K     ./.config/openbox
```

STEP 3 Now enter: `du`. You should see lots of text fly up the screen. This is the disk usage for the files contained in your home directory and their sub-directories. As with df, it is better to use du with the "-h" option to humanise the output. If you want to slow down the output, you'll also need to pipe it through less. Enter: `df -h | less` to view the files and their respective usage one page at a time.



```
pi@raspberrypi ~ $ df -h
Filesystem      Size  Used Avail Use% Mounted on
rootfs          6.3G  3.4G  2.7G  56% /
/dev/root       6.3G  3.4G  2.7G  56% /
devtmpfs        428M     0  428M   0% /dev
tmpfs            87M  260K   87M   1% /run
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           173M     0  173M   0% /run/shm
/dev/mmcblk0p5   60M   15M   45M  25% /boot
pi@raspberrypi ~ $
```

STEP 4 You don't typically enter: `du` on its own; most of the time you want to view the disk usage of a specific directory. Enter: `du -h python_games` to view how much space the python_games directory (installed alongside Raspbian) takes up. It should be 1.8M. If you want a more comprehensive breakdown of the files contained, use the "-a" option (all). Enter: `du -ha python_games` to view all the files contained and their disk usage.



```
pi@raspberrypi ~ $ ls
Desktop  Documents  Downloads  indiecity  people  python_games  Scratch  test
pi@raspberrypi ~ $ du -h python_games
1.8M    python_games
pi@raspberrypi ~ $ du -ha python_games
12K     python_games/RedSelector.png
12K     python_games/4row_board.png
12K     python_games/Star.png
28K     python_games/4row_humanwinner.png
12K     python_games/Wall_Block_Tall.png
8.0K    python_games/princess.png
12K     python_games/Selector.png
8.0K    python_games/4row_black.png
4.0K    python_games/catanimation.py
20K     python_games/flippy.py
36K     python_games/match3.wav
24K     python_games/starpusher.py
4.0K    python_games/grass1.png
40K     python_games/beep2.ogg
12K     python_games/slidepuzzle.py
8.0K    python_games/gem5.png
16K     python_games/fourinarow.py
8.0K    python_games/gem2.png
336K    python_games/flippyboard.png
12K     python_games/Grass_Block.png
```

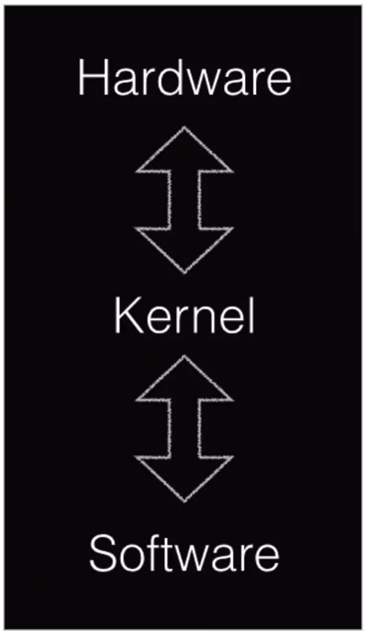
Managing Programs and Processes

Being able to effectively manage the active programs and processes on your Raspberry Pi allows you to change the way the systems work. If you have a project requiring more memory, you can kill a process to free up the available system resources.

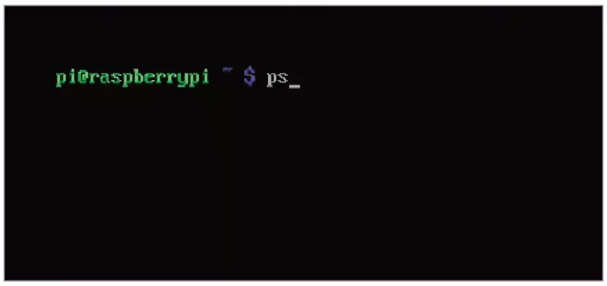
PROGRAMS AND PROCESSES

Linux has a trick up its sleeve when it comes to being able to manage programs and processes. When Windows closes a program, the allocated memory often isn't freed up again. However, Linux is far more streamlined.

STEP 1 As you get into Linux you'll start to hear more about processes and another thing called the "kernel". The kernel sits beneath the software and hardware. It passes instructions to the hardware running processes, which takes up memory, and when the process is finished it closes it and reclaims the memory.

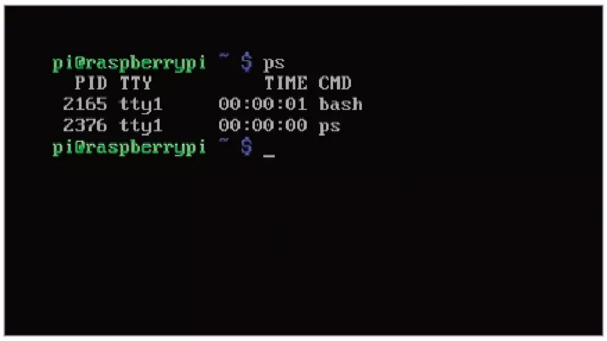


STEP 2 You're probably used to thinking in terms of programs and most OS's tend to keep processes out of sight. In Linux on the other hand, you're right in at the deep end. A process is like a program, only it's a single task running on your computer. Programs as you know them, may be a single process, or multiple processes working together. In Linux, you should learn to manage processes. This is done using the "ps" (process status) command.



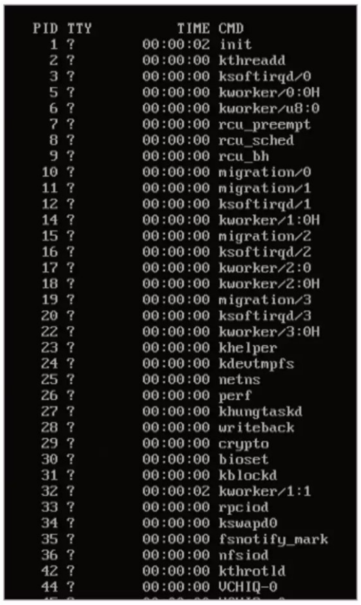
```
pi@raspberrypi ~ $ ps_
```

STEP 3 If you type in "ps" on its own you don't see much. You should see two items: bash and ps. Bash (Bourne Again Shell) is the command line environment you are typing in, and ps is the command you just entered. If that seems a little light, that's because it is. These are just the processes owned by the user and are running in the foreground.



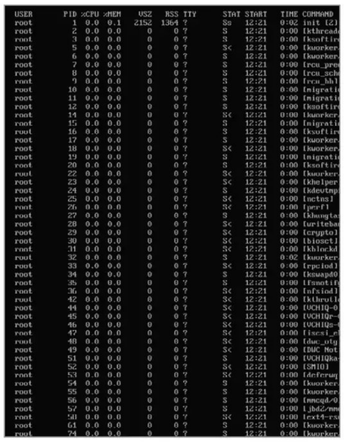
```
pi@raspberrypi ~ $ ps
  PID TTY          TIME CMD
 2165 tty1     00:00:01 bash
 2376 tty1     00:00:00 ps
pi@raspberrypi ~ $ _
```

STEP 4 If you want to see processes used by other users (including those started by root) enter: **ps –a**. The option stands for all users. This still isn't everything though because it doesn't include background processes. For this you can enter either "ps –A" or "ps –e". This will show you every process on the system including the background processes. You may need to pipe it through less using "ps -e | less".



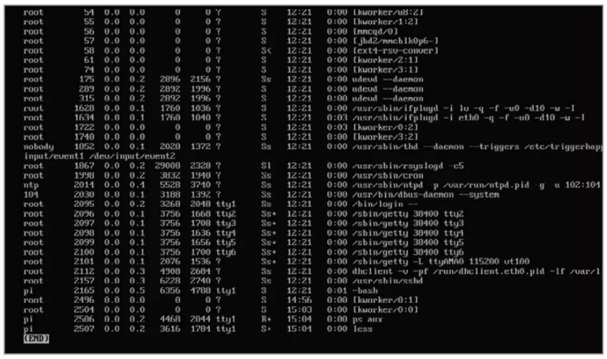
```
PID TTY          TIME CMD
  1 ?        00:00:02 init
  2 ?        00:00:00 kthreadd
  3 ?        00:00:00 ksoftirqd/0
  5 ?        00:00:00 kworker/0:0H
  6 ?        00:00:00 kworker/u8:0
  7 ?        00:00:00 rcu_preempt
  8 ?        00:00:00 rcu_sched
  9 ?        00:00:00 rcu_bh
 10 ?        00:00:00 migration/0
 11 ?        00:00:00 migration/1
 12 ?        00:00:00 ksoftirqd/1
 14 ?        00:00:00 kworker/1:0H
 15 ?        00:00:00 migration/2
 16 ?        00:00:00 ksoftirqd/2
 17 ?        00:00:00 kworker/2:0
 18 ?        00:00:00 kworker/2:0H
 19 ?        00:00:00 migration/3
 20 ?        00:00:00 ksoftirqd/3
 22 ?        00:00:00 kworker/3:0H
 23 ?        00:00:00 khelper
 24 ?        00:00:00 kdevtmpfs
 25 ?        00:00:00 netns
 26 ?        00:00:00 perf
 27 ?        00:00:00 khungtaskd
 28 ?        00:00:00 writeback
 29 ?        00:00:00 crypto
 30 ?        00:00:00 bioset
 31 ?        00:00:00 kblockd
 32 ?        00:00:02 kworker/1:1
 33 ?        00:00:00 rpciod
 34 ?        00:00:00 kswapd0
 35 ?        00:00:00 fsnotify_mark
 36 ?        00:00:00 nfsiod
 42 ?        00:00:00 kthrotld
 44 ?        00:00:00 VCHIQ-0
```

STEP 5 The ps command is one of the oldest known, and it has a lot of legacy. Because of this, it also has alternative options that can be used without the dash at the start. The most common way to use ps is to enter: `ps aux`. Piping this command through less is the best way to go about using ps. Enter: `ps aux | less` to view your processes.



```
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.1   2152  1364 ?        Ss   12:21   0:02 init [2]
root         2  0.0  0.0      0     0 ?        S    12:21   0:00 [kthread
root         3  0.0  0.0      0     0 ?        S    12:21   0:00 [ksoftir
root         5  0.0  0.0      0     0 ?        S<   12:21   0:00 [kworker
root         6  0.0  0.0      0     0 ?        S    12:21   0:00 [kworker
root         7  0.0  0.0      0     0 ?        S    12:21   0:00 [rcu_pre
root         8  0.0  0.0      0     0 ?        S    12:21   0:00 [rcu_sch
root         9  0.0  0.0      0     0 ?        S    12:21   0:00 [rcu_bh]
root        10  0.0  0.0      0     0 ?        S    12:21   0:00 [migrati
root        11  0.0  0.0      0     0 ?        S    12:21   0:00 [migrati
root        12  0.0  0.0      0     0 ?        S    12:21   0:00 [ksoftir
root        14  0.0  0.0      0     0 ?        S<   12:21   0:00 [kworker
root        15  0.0  0.0      0     0 ?        S    12:21   0:00 [migrati
root        16  0.0  0.0      0     0 ?        S    12:21   0:00 [ksoftir
root        17  0.0  0.0      0     0 ?        S    12:21   0:00 [kworker
root        18  0.0  0.0      0     0 ?        S<   12:21   0:00 [kworker
root        19  0.0  0.0      0     0 ?        S    12:21   0:00 [migrati
root        20  0.0  0.0      0     0 ?        S    12:21   0:00 [ksoftir
root        22  0.0  0.0      0     0 ?        S<   12:21   0:00 [kworker
root        23  0.0  0.0      0     0 ?        S<   12:21   0:00 [khelper
root        24  0.0  0.0      0     0 ?        S    12:21   0:00 [kdevtmp
root        25  0.0  0.0      0     0 ?        S<   12:21   0:00 [netns]
root        26  0.0  0.0      0     0 ?        S<   12:21   0:00 [perf]
root        27  0.0  0.0      0     0 ?        S    12:21   0:00 [khungta
root        28  0.0  0.0      0     0 ?        S<   12:21   0:00 [writeba
root        29  0.0  0.0      0     0 ?        S<   12:21   0:00 [crypto]
root        30  0.0  0.0      0     0 ?        S<   12:21   0:00 [biosct]
root        31  0.0  0.0      0     0 ?        S<   12:21   0:00 [kblockd
root        32  0.0  0.0      0     0 ?        S    12:21   0:02 [kworker
root        33  0.0  0.0      0     0 ?        S<   12:21   0:00 [rpciod]
root        34  0.0  0.0      0     0 ?        S    12:21   0:00 [kswapd0
root        35  0.0  0.0      0     0 ?        S    12:21   0:00 [fsnotif
root        36  0.0  0.0      0     0 ?        S<   12:21   0:00 [nfsiod]
root        42  0.0  0.0      0     0 ?        S<   12:21   0:00 [kthrotl
root        44  0.0  0.0      0     0 ?        S<   12:21   0:00 [VCHIQ-0
root        45  0.0  0.0      0     0 ?        S<   12:21   0:00 [VCHIQr-
root        46  0.0  0.0      0     0 ?        S<   12:21   0:00 [VCHIQs-
root        47  0.0  0.0      0     0 ?        S<   12:21   0:00 [iscsi_e
root        48  0.0  0.0      0     0 ?        S<   12:21   0:00 [dwc_otg
root        49  0.0  0.0      0     0 ?        S<   12:21   0:00 [DWC Not
root        51  0.0  0.0      0     0 ?        S    12:21   0:00 [VCHIQka
root        52  0.0  0.0      0     0 ?        S<   12:21   0:00 [SMIO]
root        53  0.0  0.0      0     0 ?        S<   12:21   0:00 [deferwq
root        54  0.0  0.0      0     0 ?        S    12:21   0:00 [kworker
root        55  0.0  0.0      0     0 ?        S    12:21   0:00 [kworker
root        56  0.0  0.0      0     0 ?        S    12:21   0:00 [mmcqd/0
root        57  0.0  0.0      0     0 ?        S    12:21   0:00 [jbd2/mm
root        58  0.0  0.0      0     0 ?        S<   12:21   0:00 [ext4-rs
root        61  0.0  0.0      0     0 ?        S    12:21   0:00 [kworker
root        74  0.0  0.0      0     0 ?        S    12:21   0:00 [kworker
```

STEP 6 The "ps aux" command displays the processes in columns. The first is the User followed by the Process ID (PID), then we see the percentage of CPU and memory.

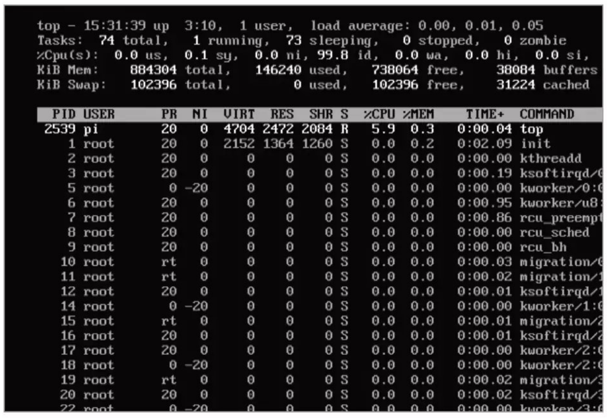


```
root        54  0.0  0.0      0     0 ?        S    12:21   0:00 [kworker/u8:2]
root        55  0.0  0.0      0     0 ?        S    12:21   0:00 [kworker/1:2]
root        56  0.0  0.0      0     0 ?        S    12:21   0:00 [mmcqd/0]
root        57  0.0  0.0      0     0 ?        S    12:21   0:00 [jbd2/mmcblk0p6-]
root        58  0.0  0.0      0     0 ?        S<   12:21   0:00 [ext4-rsv-conver]
root        61  0.0  0.0      0     0 ?        S    12:21   0:00 [kworker/2:1]
root        74  0.0  0.0      0     0 ?        S    12:21   0:00 [kworker/3:1]
root       175  0.0  0.2   2896  2156 ?        Ss   12:21   0:00 udevd --daemon
root       289  0.0  0.2   2892  1996 ?        S    12:21   0:00 udevd --daemon
root       315  0.0  0.2   2892  1996 ?        S    12:21   0:00 udevd --daemon
root      1628  0.0  0.1   1760  1036 ?        S    12:21   0:00 /usr/sbin/ifplugd -i lo -q -f -u0 -d10 -w -I
root      1634  0.0  0.1   1760  1040 ?        S    12:21   0:03 /usr/sbin/ifplugd -i eth0 -q -f -u0 -d10 -w -I
root      1722  0.0  0.0      0     0 ?        S    12:21   0:03 [kworker/0:2]
root      1740  0.0  0.0      0     0 ?        S    12:21   0:00 [kworker/3:2]
nobody    1852  0.0  0.1   2020  1372 ?        Ss   12:21   0:00 /usr/sbin/thd --daemon --triggers /etc/triggerhap
input/event1 /dev/input/event2
root      1867  0.0  0.2  29008  2328 ?        Sl   12:21   0:00 /usr/sbin/rsyslogd -c5
root      1998  0.0  0.2   3832  1940 ?        Ss   12:21   0:00 /usr/sbin/cron
ntp       2014  0.0  0.4   5528  3740 ?        Ss   12:21   0:00 /usr/sbin/ntpd -p /var/run/ntpd.pid -g -u 102:104
104       2030  0.0  0.1   3188  1392 ?        Ss   12:21   0:00 /usr/bin/dbus-daemon --system
root      2095  0.0  0.2   3268  2048 tty1     Ss   12:21   0:00 /bin/login --
root      2096  0.0  0.1   3756  1668 tty2     Ss+  12:21   0:00 /sbin/getty 38400 tty2
root      2097  0.0  0.1   3756  1708 tty3     Ss+  12:21   0:00 /sbin/getty 38400 tty3
root      2098  0.0  0.1   3756  1636 tty4     Ss+  12:21   0:00 /sbin/getty 38400 tty4
root      2099  0.0  0.1   3756  1656 tty5     Ss+  12:21   0:00 /sbin/getty 38400 tty5
root      2100  0.0  0.1   3756  1700 tty6     Ss+  12:21   0:00 /sbin/getty 38400 tty6
root      2101  0.0  0.1   2076  1536 ?        Ss+  12:21   0:00 /sbin/getty -L ttyAMA0 115200 vt100
root      2112  0.0  0.3   4908  2684 ?        Ss   12:21   0:00 dhclient -v -pf /run/dhclient.eth0.pid -lf /var/l
root      2157  0.0  0.3   6228  2740 ?        Ss   12:21   0:00 /usr/sbin/sshd
pi        2165  0.0  0.5   6356  4788 tty1     S    12:21   0:01 -bash
root      2496  0.0  0.0      0     0 ?        S    14:56   0:00 [kworker/0:1]
root      2504  0.0  0.0      0     0 ?        S    15:03   0:00 [kworker/0:0]
pi        2506  0.0  0.2   4468  2044 tty1     R+   15:04   0:00 ps aux
pi        2507  0.0  0.2   3616  1784 tty1     S+   15:04   0:00 less
(END)
```

VIEWING AND QUITTING PROCESSES

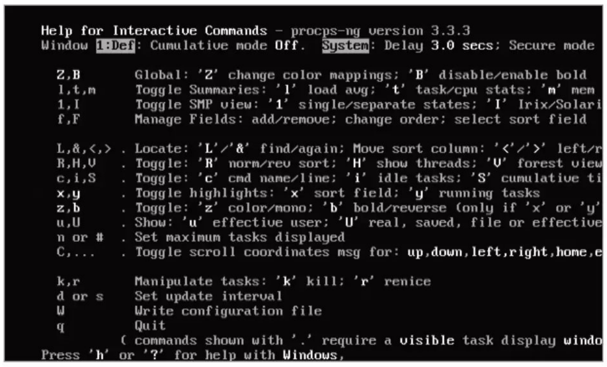
Now that you've got a handle on processes, you need to know what to do with them. You can view processes running in real-time, and quit ones that you no longer need, or those that are problematic.

STEP 1 While ps is a great tool for checking all the processes on your Raspberry Pi, sometimes you need to view them in real-time. In particular you may want to check the CPU and memory usage of processes and see if any process is hogging all your computer's resources. Use "top" to do this.



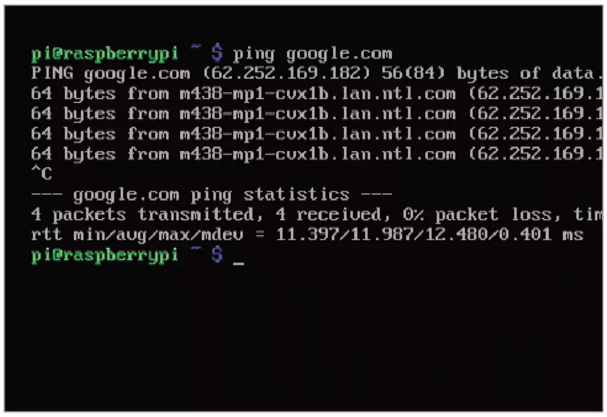
```
top - 15:31:39 up  3:10,  1 user,  load average: 0.00, 0.01, 0.05
Tasks:  74 total,   1 running,  73 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.1 sy,  0.0 ni, 99.8 id,  0.0 wa,  0.0 hi,  0.0 si,
KiB Mem:    884304 total,   146240 used,   738064 free,    38084 buffers
KiB Swap:   102396 total,        0 used,   102396 free,    31224 cached

  PID USER      PR  NI  VIRT  RES  SHR S  %CPU %MEM     TIME+  COMMAND
 2539 pi        20   0  4704 2472 2084 R   5.9  0.3   0:00.04 top
    1 root      20   0  2152 1364 1260 S   0.0  0.2   0:02.09 init
    2 root      20   0     0    0    0 S   0.0  0.0   0:00.00 kthreadd
    3 root      20   0     0    0    0 S   0.0  0.0   0:00.19 ksoftirqd/
    5 root       0 -20     0    0    0 S   0.0  0.0   0:00.00 kworker/0:
    6 root      20   0     0    0    0 S   0.0  0.0   0:00.95 kworker/u8
    7 root      20   0     0    0    0 S   0.0  0.0   0:00.86 rcu_preempt
    8 root      20   0     0    0    0 S   0.0  0.0   0:00.00 rcu_sched
    9 root      20   0     0    0    0 S   0.0  0.0   0:00.00 rcu_bh
   10 root      rt   0     0    0    0 S   0.0  0.0   0:00.03 migration/
   11 root      rt   0     0    0    0 S   0.0  0.0   0:00.02 migration/
   12 root      20   0     0    0    0 S   0.0  0.0   0:00.01 ksoftirqd/
   14 root       0 -20     0    0    0 S   0.0  0.0   0:00.00 kworker/1:
   15 root      rt   0     0    0    0 S   0.0  0.0   0:00.01 migration/
   16 root      20   0     0    0    0 S   0.0  0.0   0:00.01 ksoftirqd/
   17 root      20   0     0    0    0 S   0.0  0.0   0:00.00 kworker/2:
   18 root       0 -20     0    0    0 S   0.0  0.0   0:00.00 kworker/2:
   19 root      rt   0     0    0    0 S   0.0  0.0   0:00.02 migration/
   20 root      20   0     0    0    0 S   0.0  0.0   0:00.02 ksoftirqd/
   22 root       0 -20     0    0    0 S   0.0  0.0   0:00.00 kworker/3:
```

STEP 2 Top fills the display with processes and it fits as many as it can on one screen. These run in real-time so as you watch, the display items will move up and down according to their usage. You can interact with top as it is running: use < and > to change the sort column. Press H to view a help page with all the commands you can use.



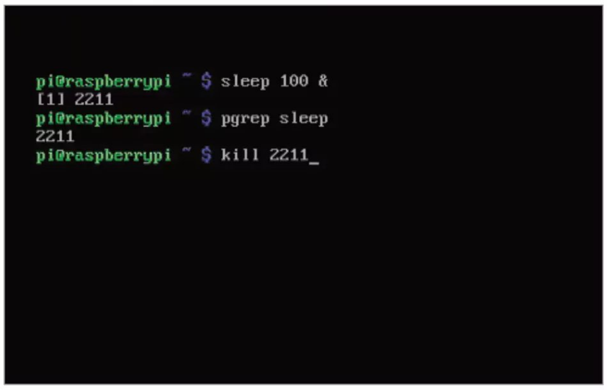
```
Help for Interactive Commands - procps-ng version 3.3.3
Window 1:Def: Cumulative mode Off.  System: Delay 3.0 secs; Secure mode

  Z,B       Global: 'Z' change color mappings; 'B' disable/enable bold
  l,t,m     Toggle Summaries: 'l' load avg; 't' task/cpu stats; 'm' mem
  1,I       Toggle SMP view: '1' single/separate states; 'I' Irix/Solari
  f,F       Manage Fields: add/remove; change order; select sort field

  L,&,<,> . Locate: 'L'/'&' find/again; Move sort column: '<'/'>' left/r
  R,H,V   . Toggle: 'R' norm/rev sort; 'H' show threads; 'V' forest view
  c,i,S   . Toggle: 'c' cmd name/line; 'i' idle tasks; 'S' cumulative ti
  x,y     . Toggle highlights: 'x' sort field; 'y' running tasks
  z,b     . Toggle: 'z' color/mono; 'b' bold/reverse (only if 'x' or 'y'
  u,U     . Show: 'u' effective user; 'U' real, saved, file or effective
  n or #  . Set maximum tasks displayed
  C,...   . Toggle scroll coordinates msg for: up,down,left,right,home,e

  k,r       Manipulate tasks: 'k' kill; 'r' renice
  d or s    Set update interval
  W         Write configuration file
  q         Quit
          ( commands shown with '.' require a visible task display windo
Press 'h' or '?' for help with Windows,
```

STEP 3 Now that you've got an idea of what processes are, how do you go about stopping them? Some processes run in the foreground until you quit them. Enter: `ping www.google.com` and you'll start a ping process continuously contacting Google. This sort of process runs until you press Control+C.



```
pi@raspberrypi ~ $ ping google.com
PING google.com (62.252.169.182) 56(84) bytes of data.
64 bytes from m438-mp1-cvx1b.lan.ntl.com (62.252.169.1
64 bytes from m438-mp1-cvx1b.lan.ntl.com (62.252.169.1
64 bytes from m438-mp1-cvx1b.lan.ntl.com (62.252.169.1
64 bytes from m438-mp1-cvx1b.lan.ntl.com (62.252.169.1
^C
--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, tim
rtt min/avg/max/mdev = 11.397/11.987/12.480/0.401 ms
pi@raspberrypi ~ $ _
```

STEP 4 Processes are stopped using the kill command. Enter: `sleep 100 &` to create a dummy process. Enter: `pgrep sleep` to find its PID number (on ours it is 2192). Now enter: `kill 2192` and the sleep process will be stopped. You can also use `pkill sleep` if you're sure you know the process name.



```
pi@raspberrypi ~ $ sleep 100 &
[1] 2211
pi@raspberrypi ~ $ pgrep sleep
2211
pi@raspberrypi ~ $ kill 2211_
```

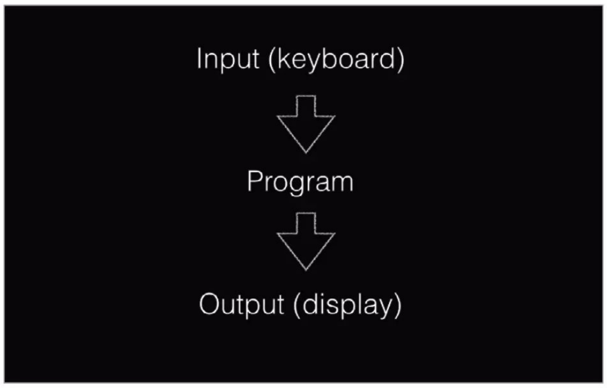
Input, Output and Pipes

Most operating systems allow you to direct the output of something on the screen to, for example, a file. Linux, with its ancestral history of Unix-like command-based arguments, goes a step further and offers much more control.

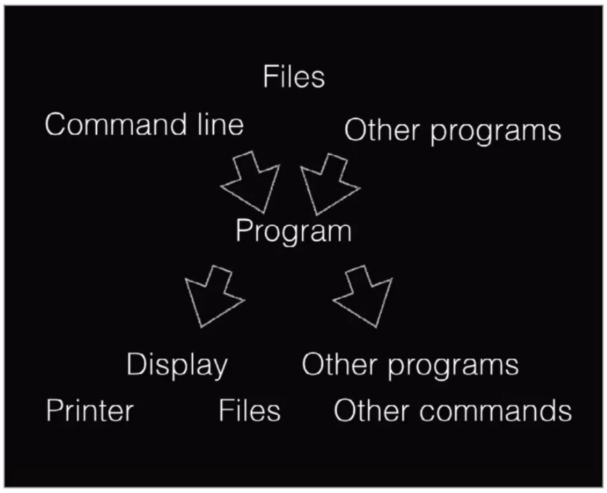
IN, OUT, LINUX ALL ABOUT

Everything on a computer is about input and output. You input things into the computer (press keys, move the mouse) and it makes calculations and outputs content (changes the display, makes a noise, for example).

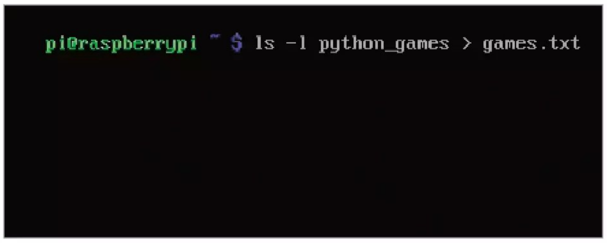
STEP 1 When you enter commands into Linux (in the command line), you are using standard input and output. This is so obvious that most people don't even think about it. You enter commands using the keyboard (that's input) and it responds to the screen (output). This is the regular way of doing it, which is why it's "standard input and output" (often called "stdin" and "stdout" for short).



STEP 2 As far as the computer is concerned, input and output can be to and from a whole range of different sources that you might never even think about. A program can get input from other programs, from files stored on the hard drive and a whole host of other areas. It outputs back to the display line, but also to files, other programs and even other commands.

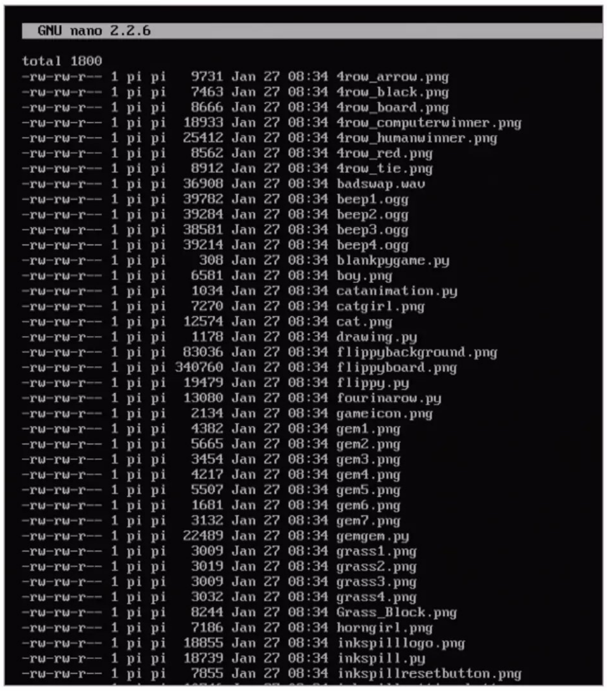


STEP 3 You can change the standard output to something else using the ">" character after your command. If we wanted to see all the items in the python_games directory we could use "ls -l python_games". Using "ls -l python_games > games.txt" outputs the list of items to a new text file called "games.txt".



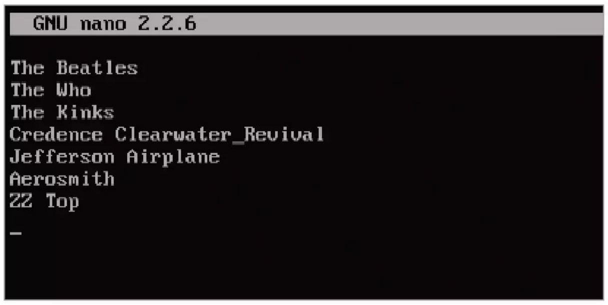
```
pi@raspberrypi ~ $ ls -l python_games > games.txt
```

STEP 4 The games.txt file now contains the output from the ls command. You can check it using "nano games.txt". It's an editable text file containing all the permissions, user and file size information and the names of the files. The output from the ls -l command, normally displayed on the screen, was instead sent to this file. Press Control+X to quit nano.



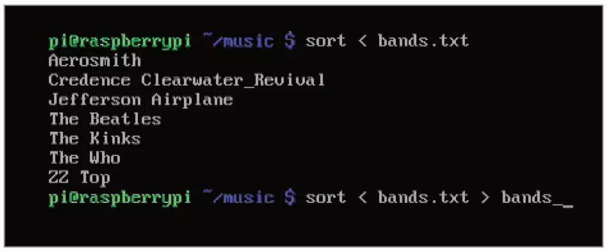
```
  GNU nano 2.2.6

total 1800
-rw-rw-r-- 1 pi pi   9731 Jan 27 08:34 4row_arrow.png
-rw-rw-r-- 1 pi pi   7463 Jan 27 08:34 4row_black.png
-rw-rw-r-- 1 pi pi   8666 Jan 27 08:34 4row_board.png
-rw-rw-r-- 1 pi pi  18933 Jan 27 08:34 4row_computerwinner.png
-rw-rw-r-- 1 pi pi  25412 Jan 27 08:34 4row_humanwinner.png
-rw-rw-r-- 1 pi pi   8562 Jan 27 08:34 4row_red.png
-rw-rw-r-- 1 pi pi   8912 Jan 27 08:34 4row_tie.png
-rw-rw-r-- 1 pi pi  36908 Jan 27 08:34 badswap.wav
-rw-rw-r-- 1 pi pi  39782 Jan 27 08:34 beep1.ogg
-rw-rw-r-- 1 pi pi  39284 Jan 27 08:34 beep2.ogg
-rw-rw-r-- 1 pi pi  38581 Jan 27 08:34 beep3.ogg
-rw-rw-r-- 1 pi pi  39214 Jan 27 08:34 beep4.ogg
-rw-rw-r-- 1 pi pi    308 Jan 27 08:34 blankpygame.py
-rw-rw-r-- 1 pi pi   6581 Jan 27 08:34 boy.png
-rw-rw-r-- 1 pi pi   1034 Jan 27 08:34 catanimation.py
-rw-rw-r-- 1 pi pi   7270 Jan 27 08:34 catgirl.png
-rw-rw-r-- 1 pi pi  12574 Jan 27 08:34 cat.png
-rw-rw-r-- 1 pi pi   1178 Jan 27 08:34 drawing.py
-rw-rw-r-- 1 pi pi  83036 Jan 27 08:34 flippybackground.png
-rw-rw-r-- 1 pi pi 340760 Jan 27 08:34 flippyboard.png
-rw-rw-r-- 1 pi pi  19479 Jan 27 08:34 flippy.py
-rw-rw-r-- 1 pi pi  13080 Jan 27 08:34 fourinarow.py
-rw-rw-r-- 1 pi pi   2134 Jan 27 08:34 gameicon.png
-rw-rw-r-- 1 pi pi   4382 Jan 27 08:34 gem1.png
-rw-rw-r-- 1 pi pi   5665 Jan 27 08:34 gem2.png
-rw-rw-r-- 1 pi pi   3454 Jan 27 08:34 gem3.png
-rw-rw-r-- 1 pi pi   4217 Jan 27 08:34 gem4.png
-rw-rw-r-- 1 pi pi   5507 Jan 27 08:34 gem5.png
-rw-rw-r-- 1 pi pi   1681 Jan 27 08:34 gem6.png
-rw-rw-r-- 1 pi pi   3132 Jan 27 08:34 gem7.png
-rw-rw-r-- 1 pi pi  22489 Jan 27 08:34 gemgem.py
-rw-rw-r-- 1 pi pi   3009 Jan 27 08:34 grass1.png
-rw-rw-r-- 1 pi pi   3019 Jan 27 08:34 grass2.png
-rw-rw-r-- 1 pi pi   3009 Jan 27 08:34 grass3.png
-rw-rw-r-- 1 pi pi   3032 Jan 27 08:34 grass4.png
-rw-rw-r-- 1 pi pi   8244 Jan 27 08:34 Grass_Block.png
-rw-rw-r-- 1 pi pi   7186 Jan 27 08:34 horngirl.png
-rw-rw-r-- 1 pi pi  18855 Jan 27 08:34 inkspilllogo.png
-rw-rw-r-- 1 pi pi  18739 Jan 27 08:34 inkspill.py
-rw-rw-r-- 1 pi pi   7855 Jan 27 08:34 inkspillresetbutton.png
```

STEP 5 So > enables you to output to files, but you can also get input from a file. Make a new directory called music ("mkdir music") and switch to it ("cd music"). Enter: `nano bands.txt` to create a new text file. Enter some band names and press Control+O to output the file. Press Control+X to quit nano.



```
 GNU nano 2.2.6

The Beatles
The Who
The Kinks
Credence Clearwater_Revival
Jefferson Airplane
Aerosmith
ZZ Top
_
```

STEP 6 We're going to use this text file as input to the sort command. Enter: `sort < bands.txt` and the content from the text file is used as input to sort. Because the output isn't specified, it uses the standard output (the screen) but you use input and output together. Enter: `sort < bands.txt > bands_sorted.txt` to create a new file with the band names in order to switch back to your pi account.

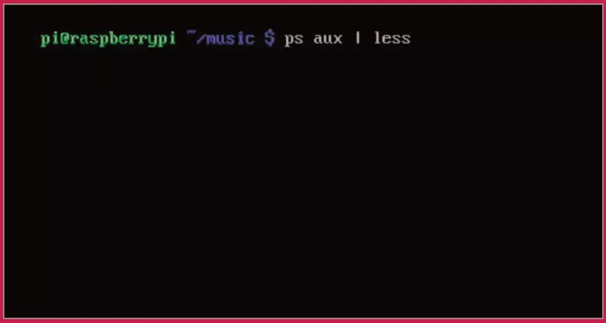


```
pi@raspberrypi ~/music $ sort < bands.txt
Aerosmith
Credence Clearwater_Revival
Jefferson Airplane
The Beatles
The Kinks
The Who
ZZ Top
pi@raspberrypi ~/music $ sort < bands.txt > bands_
```

USING PIPES

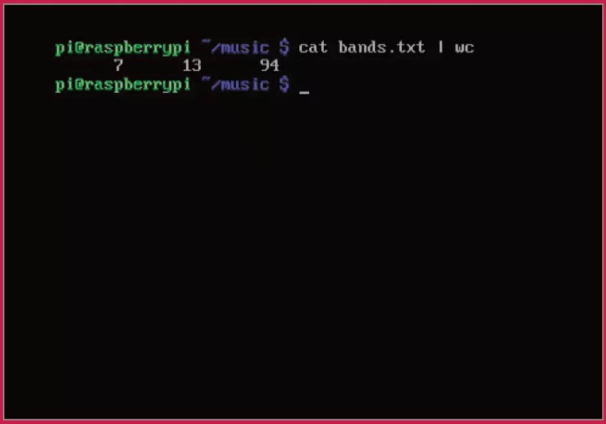
As well as directing input and output to and from files, you can send the output from one command directly into another. This is known as piping, and uses the pipe character "|".

STEP 1 As you start to get more advanced in Linux, you begin to create more powerful commands, and one way you do this is by using the pipe character ("|"). Take some time to find this character if you haven't already: it usually sits above or to the left of the Return key on most keyboards.



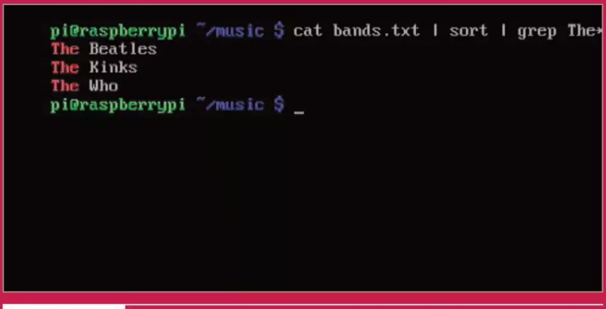
```
pi@raspberrypi ~/music $ ps aux | less
```

STEP 2 We've used the pipe a few times in the book ("ps aux | less"), but you might not have understood what's actually happening. Enter: `cat bands.txt | wc`. The output from the cat command (the text inside the document) isn't displayed on the screen, instead it is piped into the wc (word count) function. This then tells us how many lines, words and characters are in the document.



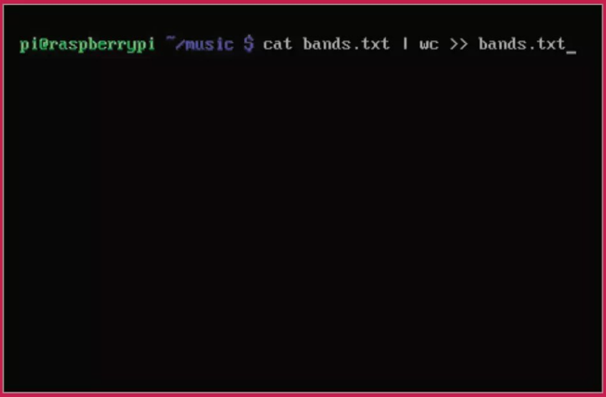
```
pi@raspberrypi ~/music $ cat bands.txt | wc
      7      13      94
pi@raspberrypi ~/music $ _
```

STEP 3 You can pipe commands multiple times. Enter: `cat bands.txt | sort | grep The*` to get the bands starting with "The" in alphabetical order. The output of the text from the bands.txt document is passed into sort, and the output from sort is passed into grep which filters out the bands starting with "The". These bands form the output.



```
pi@raspberrypi ~/music $ cat bands.txt | sort | grep The*
The Beatles
The Kinks
The Who
pi@raspberrypi ~/music $ _
```

STEP 4 You can combine pipes with input and output to create complex expressions. You can also use >> to append outputted data to a file that already exists. Enter: `cat bands.txt | wc >> bands.txt`. This takes the output from the bands.txt file and pipes it into the wc (word count) function. The output from wc is appended to the end of the bands.txt file. Enter: `cat bands.txt` to view it.



```
pi@raspberrypi ~/music $ cat bands.txt | wc >> bands.txt_
```

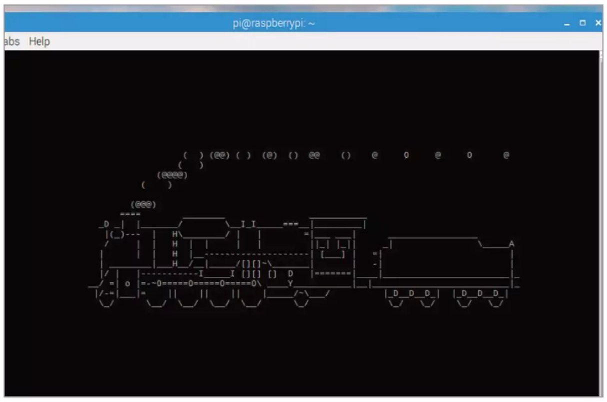
Fun Things to Do in the Terminal

Despite the seriousness of an operating system, the Linux community are certainly no strangers to a bit of fun. Over the years, the developers have created and inserted all manner of quirky and entertaining elements into the Terminal.

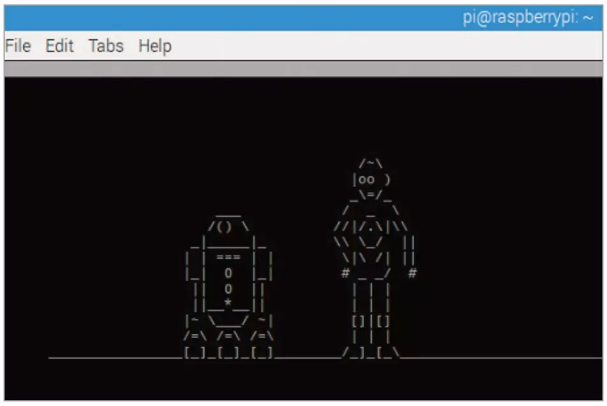
TERMINAL FUN

All these commands are Linux-based, so not only can you use them on the Raspberry Pi but also on any of the Debian-based Linux distributions.

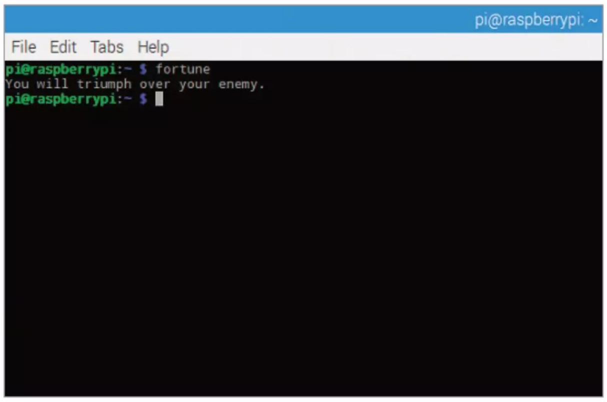
STEP 1 The first command we're going to use is `sl`, it's not installed by default so enter: `sudo apt-get install sl`. The command can be run with `sl` and when executed displays a Steam Locomotive travelling across the screen (hence 'sl'). Entering: `LS` (note the upper case) also works.



STEP 2 Fans of Star Wars even get a fix when it comes to the Terminal. By linking to a remote server via the telnet command, you can watch *Episode IV: A New Hope* being played out, albeit in ASCII. To view this spectacle, enter: `sudo apt-get install telnet`, followed by:
`telnet towel.blinkenlights.nl`

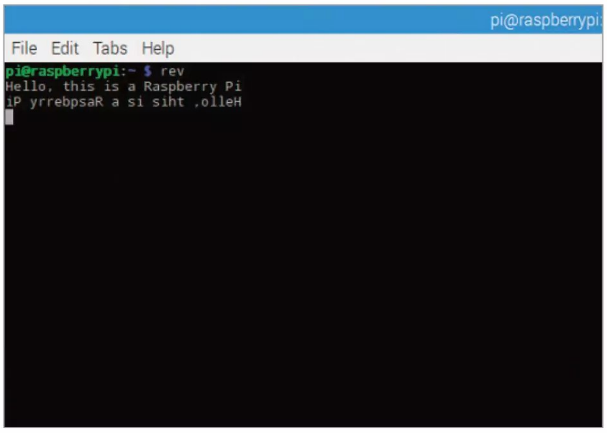


STEP 3 If you've ever fancied having the computer read a random fortune out to you then you're in luck. Raspbian requires you to install the Terminal app, Fortune, first. Enter: `sudo apt-get install fortune`, then simply enter: `fortune`, into the Terminal to see what comes up.



```
pi@raspberrypi:~ $ fortune
You will triumph over your enemy.
pi@raspberrypi:~ $
```

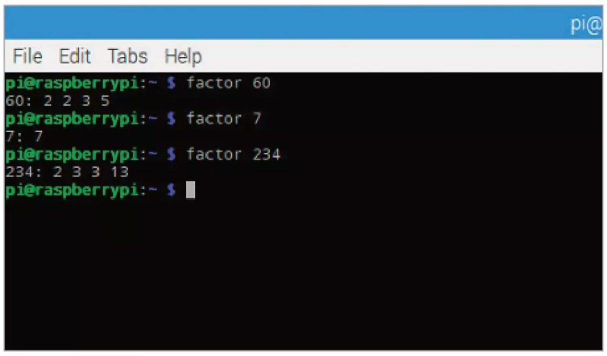
STEP 4 The rev command is certainly interesting, and at first what seems a quite useless addition to the OS can, however, be used to create some seemingly unbreakable passwords. Enter: `rev` and then type some text. Then press Enter and everything you typed in is reversed. Press Ctrl+C to exit.



```
pi@raspberrypi:~ $ rev
Hello, this is a Raspberry Pi
iP yrrebpsaR a si siht ,olleH
```

STEP 5

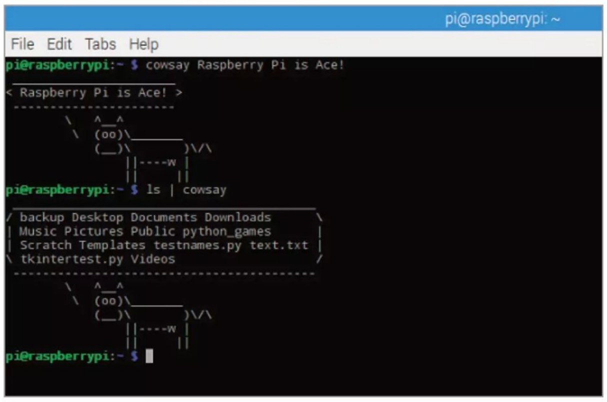
If you're stuck trying to work out all the possible factors for any particular number, simply enter: `factor` followed by the number. For example, factor 7 doesn't offer much output, whereas factor 60 displays more.



```
pi@raspberrypi:~ $ factor 60
60: 2 2 3 5
pi@raspberrypi:~ $ factor 7
7: 7
pi@raspberrypi:~ $ factor 234
234: 2 3 3 13
pi@raspberrypi:~ $
```

STEP 6

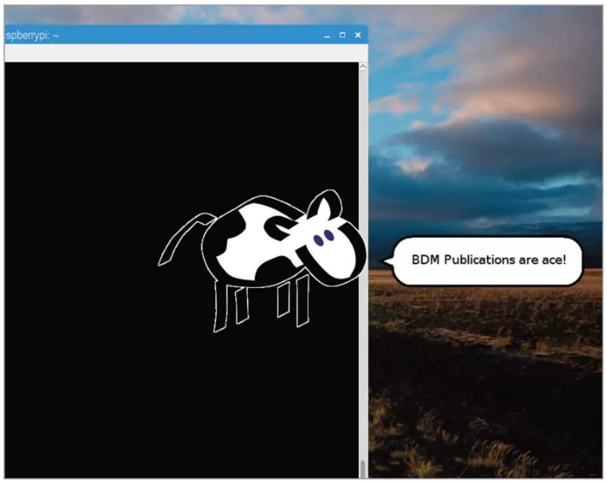
There's a fine line between the rather cool and the really quite weird. Having an ASCII cow repeat text to you could potentially fall into the latter. Start by installing cowsay: `sudo apt-get install cowsay`, then enter: `cowsay Raspberry Pi is Ace!`. In fact, you can even output the ls command through the cow, by entering: `ls | cowsay`.



```
pi@raspberrypi:~ $ cowsay Raspberry Pi is Ace!
 ______________________
< Raspberry Pi is Ace! >
 ----------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
pi@raspberrypi:~ $ ls | cowsay
 ________________________________________
/ backup Desktop Documents Downloads      \
| Music Pictures Public python_games      |
| Scratch Templates testnames.py text.txt |
\ tkintertest.py Videos                   /
 -----------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
pi@raspberrypi:~ $
```

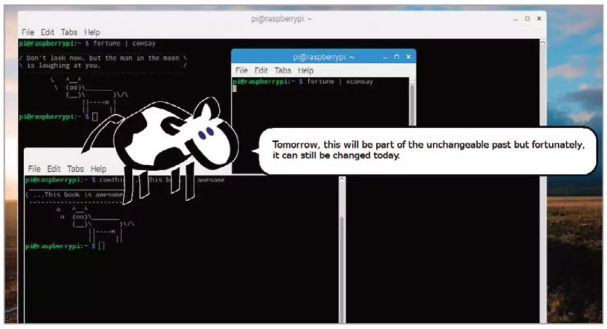
STEP 7

To further the cow element, there's even a graphical, i.e. non-Terminal, cow available. Install it with: `sudo apt-get install xcowsay`, then when it's installed enter something similar to cowsay, such as: `xcowsay BDM Publications are ace!`.



STEP 8

If you really want to expand the whole cow thing, for whatever reason, then pipe the fortune command through it, with: `fortune | cowsay`; and for the graphical cow equivalent: `fortune | xcowsay`. Plus, there's always cowthink. Try: `cowthink ...This book is awesome`.



```
pi@raspberrypi:~ $ fortune | cowsay
 _______________________________________
/ Don't look now, but the man in the moon \
\ is laughing at you.                     /
 ---------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
pi@raspberrypi:~ $
```

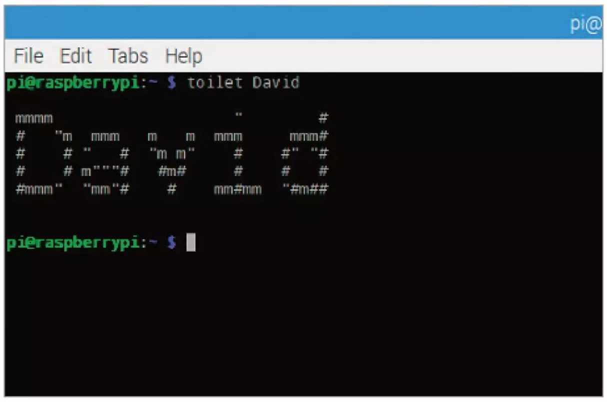
```
pi@raspberrypi:~ $ fortune | xcowsay
```

Tomorrow, this will be part of the unchangeable past but fortunately, it can still be changed today.

```
pi@raspberrypi:~ $ cowthink ...This book is awesome
 _______________________
( ...This book is awesome )
 -----------------------
        o   ^__^
         o  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
pi@raspberrypi:~ $
```

STEP 9

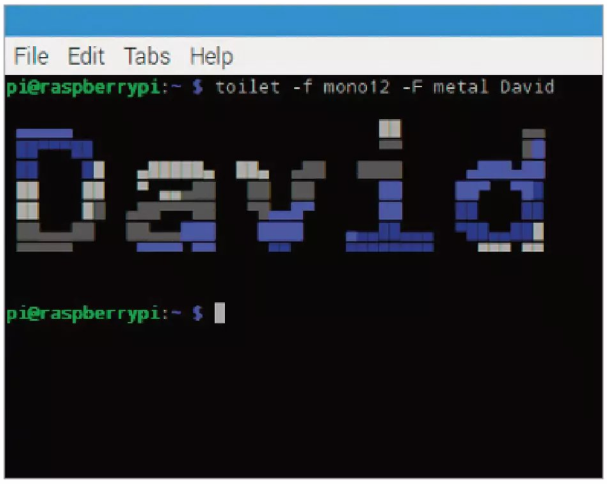
Admittedly, the command 'toilet' doesn't inspire much confidence. However, it's not as bad as it first sounds. Start by installing it with: `sudo apt-get install toilet`, then when it's installed, type something along the lines of: `toilet David`; or perhaps list the contents of the current folder through it, with: `ls | toilet`.



```
pi@raspberrypi:~ $ toilet David

 mmmm                         "         #
 #   "m  mmm   m    m  mmm      mmm#
 #    # "   #  "m  m"    #     #" "#
 #    # m"""#   #m#      #     #   #
 #mmm"  "mm"#    #     mm#mm   "#m##


pi@raspberrypi:~ $
```

STEP 10

Expanding the toilet command, you can actually generate some decent looking graphics through it. For example, try this: `toilet -f mono12 -F metal David`. You can enter: `toilet --help`, for a list of the command line arguments to expand further.



```
pi@raspberrypi:~ $ toilet -f mono12 -F metal David
```

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

More Fun Things to Do in the Terminal

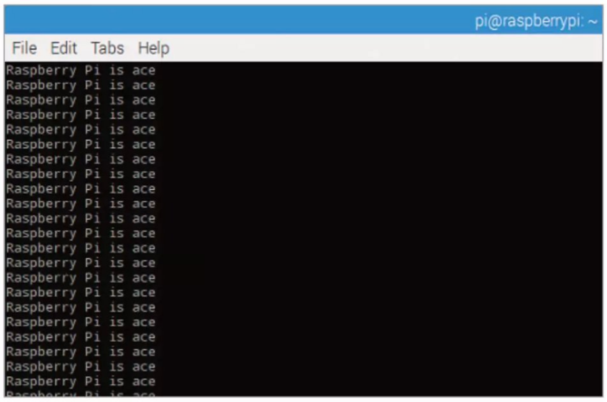
If the previous list of bizarre, and fun things to do in the Terminal has you wanting more, you're in luck. We've put together another batch of both useful, and not so useful, commands for you to try out.

MORE FUN, YAY

Since the Terminal session is already open, and your keyboard digits are nicely warmed up, here are another two pages of Terminal nonsense.

STEP 1

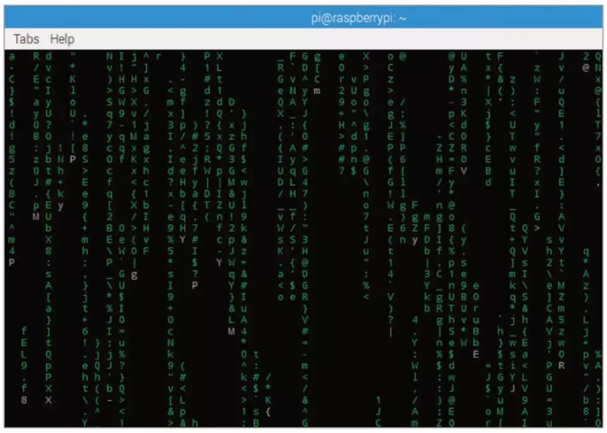
Remember the old ZX Spectrum days of computing, when you could type in 10 print "Hello", 20 goto 10 and Hello would list down the screen? Well, in Raspbian you can do the same. Simply enter: `yes` followed by some text, i.e. `yes Raspberry Pi` is ace. It keeps going until you press Ctrl+C.



```
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
Raspberry Pi is ace
```

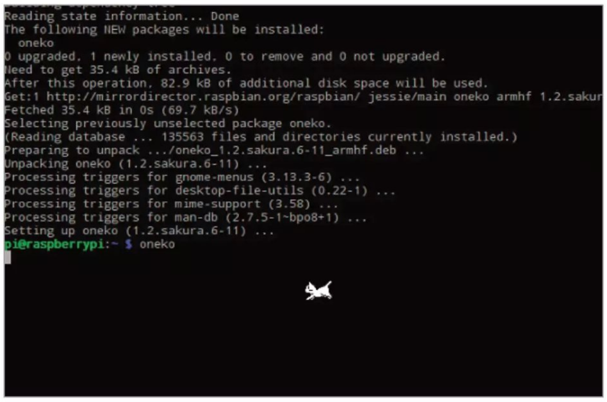
STEP 2

The Matrix was one of the most visually copied films ever released and there's even a version of the Matrix code available for Linux. Install it with: `sudo apt-get install cmatrix`. When it's done enter: `cmatrix` and follow the white rabbit, Neo. Unlike the real Matrix though, you can press Ctrl+C to exit.



STEP 3

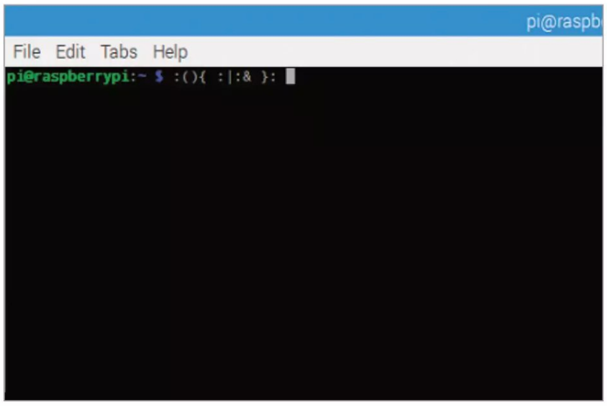
Having a little white cat chase your mouse pointer around the desktop may sound like a terrible waste of time. Oddly though, it isn't. Enter: `sudo apt-get install oneko`, then type: `oneko` to have the cat appear. Move your 'mouse' cursor around the screen and the cat chases it. Use Ctrl+C to exit the action.



```
Reading state information... Done
The following NEW packages will be installed:
  oneko
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 35.4 kB of archives.
After this operation, 82.9 kB of additional disk space will be used.
Get:1 http://mirrordirector.raspbian.org/raspbian/ jessie/main oneko armhf 1.2.sakur
Fetched 35.4 kB in 0s (69.7 kB/s)
Selecting previously unselected package oneko.
(Reading database ... 135563 files and directories currently installed.)
Preparing to unpack .../oneko_1.2.sakura.6-11_armhf.deb ...
Unpacking oneko (1.2.sakura.6-11) ...
Processing triggers for gnome-menus (3.13.3-6) ...
Processing triggers for desktop-file-utils (0.22-1) ...
Processing triggers for mime-support (3.58) ...
Processing triggers for man-db (2.7.5-1~bpo8+1) ...
Setting up oneko (1.2.sakura.6-11) ...
pi@raspberrypi:~ $ oneko
```

STEP 4

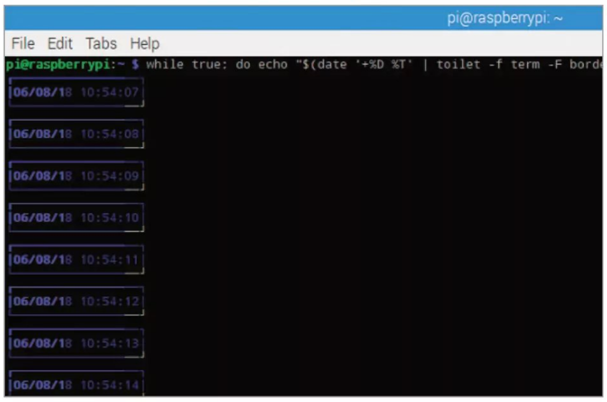
This entry is a little more serious than the previous. Fork Bomb is a command that continually replicates itself until it has used up all the available system resources, eventually causing your computer to crash. You don't have to try it, but it's interesting nonetheless. Simply enter: `:(){ :|:& };:` and be prepared to reboot.



```
pi@raspberrypi:~ $ :(){ :|:& };:
```

STEP 5

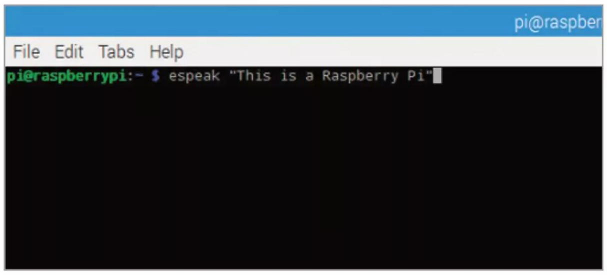
Stringing several commands and piping them through other commands is what makes scripting such a powerful element to an OS. For example, using the while command, together with toilet, can yield some impressive results. Enter: `while true; do echo "$(date '+%D %T' | toilet -f term -F border --metal)"; sleep 1; done`.



```
pi@raspberrypi: ~
File  Edit  Tabs  Help
pi@raspberrypi:~ $ while true; do echo "$(date '+%D %T' | toilet -f term -F borde
|06/08/18 10:54:07|
|06/08/18 10:54:08|
|06/08/18 10:54:09|
|06/08/18 10:54:10|
|06/08/18 10:54:11|
|06/08/18 10:54:12|
|06/08/18 10:54:13|
|06/08/18 10:54:14|
```

STEP 6

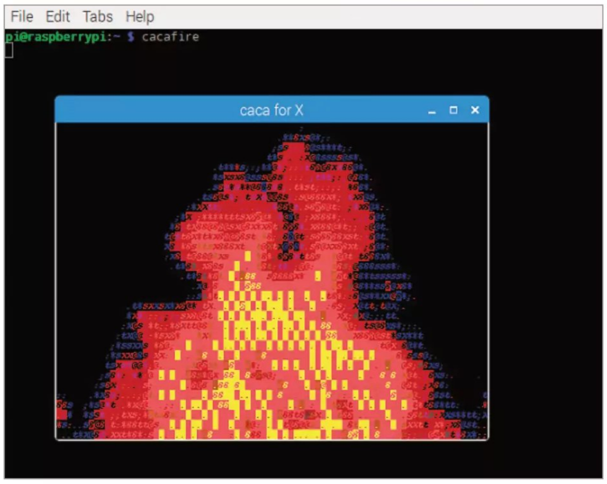
Talking computers were the craze of the '80s. To re-live the fun enter: `sudo apt-get install espeak`, then: `espeak "This is a Raspberry Pi"` to have the computer repeat the text inside the quotes to you. Make sure your volume is turned up and try the following: `ls > folders.txt && espeak -f folders.txt`. This gets Raspbian to read back the contents of the ls command.



```
pi@raspber
File  Edit  Tabs  Help
pi@raspberrypi:~ $ espeak "This is a Raspberry Pi"
```

STEP 7

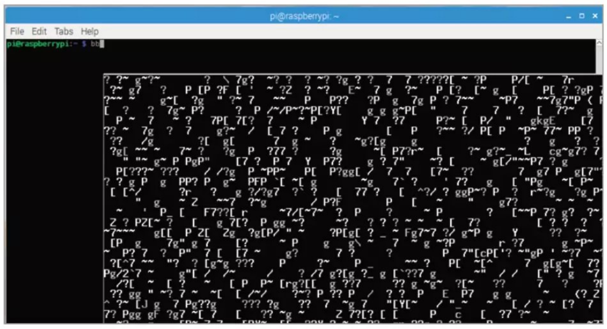
A roaring ASCII fire isn't the most useful command to have at your disposal but it's fun. Install it with: `sudo apt-get install libaa-bin`, then use: `aafire`. It's not exactly warming but you get the idea. To expand the above, enter: `sudo apt-get install bb caca-utils`, then: `cacafire`.



```
File  Edit  Tabs  Help
pi@raspberrypi:~ $ cacafire
```

STEP 8

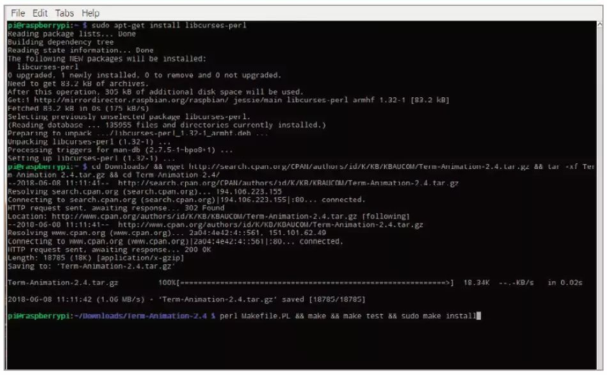
Used as a music demo from the old Amiga and DOS days, the bb command evokes memories of three and a half inch floppies crammed with all manner of demo scene goodies. You've already installed bb from the previous step, so just enter: `bb`. Follow the onscreen instructions and turn up your volume.



```
pi@raspberrypi: ~
File  Edit  Tabs  Help
pi@raspberrypi:~ $ bb
```

STEP 9

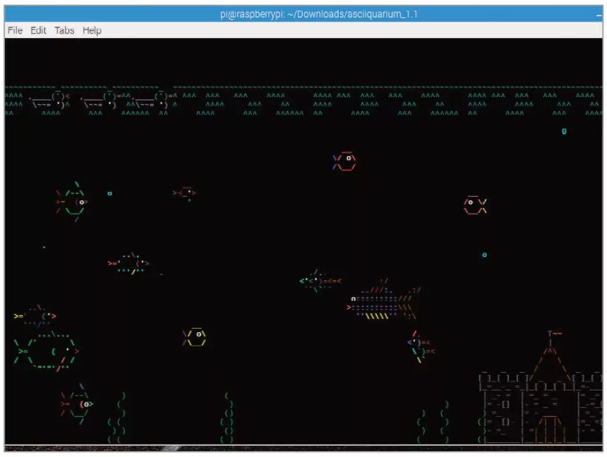
This entry is in two parts. First you need to get hold of the necessary packages: sudo apt-get install libcurses-perl. When that's done enter: `cd Downloads/ && wget http://search.cpan.org/CPAN/authors/id/K/KB/KBAUCOM/Term-Animation-2.4.tar.gz && tar -xf Term-Animation-2.4.tar.gz && cd Term-Animation-2.4/`. Followed by: `perl Makefile.PL && make && make test && sudo make install`.



```
File  Edit  Tabs  Help
pi@raspberrypi:~ $ sudo apt-get install libcurses-perl
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  libcurses-perl
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 83.2 kB of archives.
After this operation, 305 kB of additional disk space will be used.
Get:1 http://mirrordirector.raspbian.org/raspbian/ jessie/main libcurses-perl armhf 1.32-1 [83.2 kB]
Fetched 83.2 kB in 0s (175 kB/s)
Selecting previously unselected package libcurses-perl.
(Reading database ... 135955 files and directories currently installed.)
Preparing to unpack .../libcurses-perl_1.32-1_armhf.deb ...
Unpacking libcurses-perl (1.32-1) ...
Processing triggers for man-db (2.7.5-1-bpo8+1) ...
Setting up libcurses-perl (1.32-1) ...
pi@raspberrypi:~ $ cd Downloads/ && wget http://search.cpan.org/CPAN/authors/id/K/KB/KBAUCOM/Term-Animation-2.4.tar.gz && tar -xf Ter
m-Animation-2.4.tar.gz && cd Term-Animation-2.4/
--2018-06-08 11:11:41--  http://search.cpan.org/CPAN/authors/id/K/KB/KBAUCOM/Term-Animation-2.4.tar.gz
Resolving search.cpan.org (search.cpan.org)... 194.106.223.155
Connecting to search.cpan.org (search.cpan.org)|194.106.223.155|:80... connected.
HTTP request sent, awaiting response... 302 Found
Location: http://www.cpan.org/authors/id/K/KB/KBAUCOM/Term-Animation-2.4.tar.gz [following]
--2018-06-08 11:11:41--  http://www.cpan.org/authors/id/K/KB/KBAUCOM/Term-Animation-2.4.tar.gz
Resolving www.cpan.org (www.cpan.org)... 2a04:4e42:4::561, 151.101.62.49
Connecting to www.cpan.org (www.cpan.org)|2a04:4e42:4::561|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 18785 (18K) [application/x-gzip]
Saving to: 'Term-Animation-2.4.tar.gz'

Term-Animation-2.4.tar.gz       100%[===================================================================>]  18.34K  --.-KB/s   in 0.02s

2018-06-08 11:11:42 (1.06 MB/s) - 'Term-Animation-2.4.tar.gz' saved [18785/18785]

pi@raspberrypi:~/Downloads/Term-Animation-2.4 $ perl Makefile.PL && make && make test && sudo make install
```

STEP 10

With that little lot completed, onto the next part. Enter: `cd .. && wget http://www.robobunny.com/projects/asciiquarium/asciiquarium.tar.gz && tar -xf asciiquarium.tar.gz && cd asciiquarium_1.1/ && chmod +x asciiquarium`. Providing all went well, enter: `./asciiquarium` and enjoy your very own ASCII-based aquarium.



```
pi@raspberrypi: ~/Downloads/asciiquarium_1.1
File  Edit  Tabs  Help
```

Linux Tips and Tricks

The Linux Terminal, you'll no doubt agree, is an exceptional environment and with a few extra apps installed along with a smidgen of command knowledge, incredible and often quite strange things can be accomplished.

TAKING COMMAND

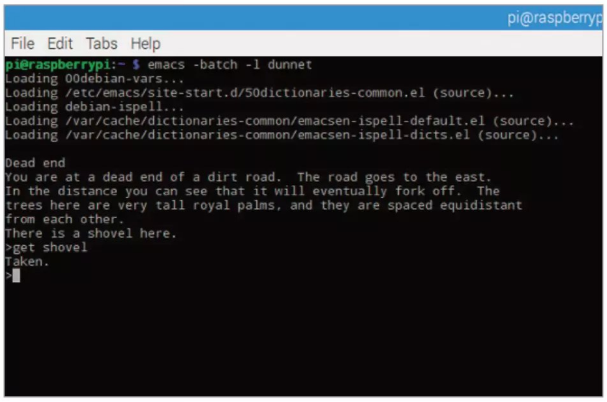
There are countless Linux tips, secrets, hacks and tricks out there. Some are very old, originating from Linux's Unix heritage, while others are recent additions to Linux lore. Here are our ten favourite tips and tricks.

EASTER EGGS

Emacs text editor, is a great piece of software but did you know it also contains a hidden Easter Egg? With Emacs installed (sudo apt-get install emacs24), drop to a Terminal session and enter:

```
emacs -batch -l dunnet
```

Dunnet is a text adventure written by Ron Schnell in 1982, and hidden in Emacs since 1994.



```
pi@raspberrypi:~ $ emacs -batch -l dunnet
Loading 00debian-vars...
Loading /etc/emacs/site-start.d/50dictionaries-common.el (source)...
Loading debian-ispell...
Loading /var/cache/dictionaries-common/emacsen-ispell-default.el (source)...
Loading /var/cache/dictionaries-common/emacsen-ispell-dicts.el (source)...

Dead end
You are at a dead end of a dirt road.  The road goes to the east.
In the distance you can see that it will eventually fork off.  The
trees here are very tall royal palms, and they are spaced equidistant
from each other.
There is a shovel here.
>get shovel
Taken.
>
```

TERMINAL BROWSING

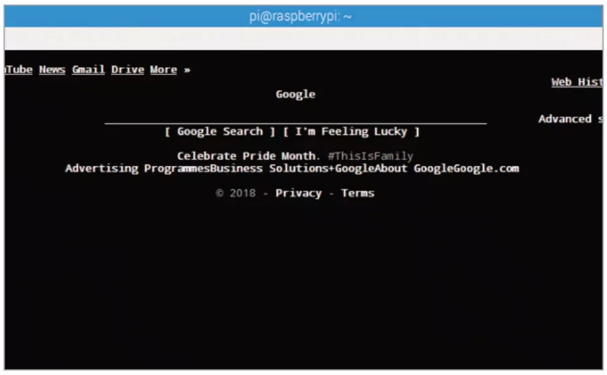
Ever fancied being able to browse the Internet from the Terminal? While not particularly useful, it is a fascinating thing to behold. To do so, enter:

```
sudo apt-get install elinks
```

Then:

```
elinks
```

Enter the website you want to visit.



```
Tube News Gmail Drive More »
                                                                    Web Hist
                                Google
                                                                  Advanced s
          ________________________________________________
               [ Google Search ] [ I'm Feeling Lucky ]

                  Celebrate Pride Month. #ThisIsFamily
     Advertising ProgrammesBusiness Solutions+GoogleAbout GoogleGoogle.com

                         © 2018 - Privacy - Terms
```

MOON BUGGY

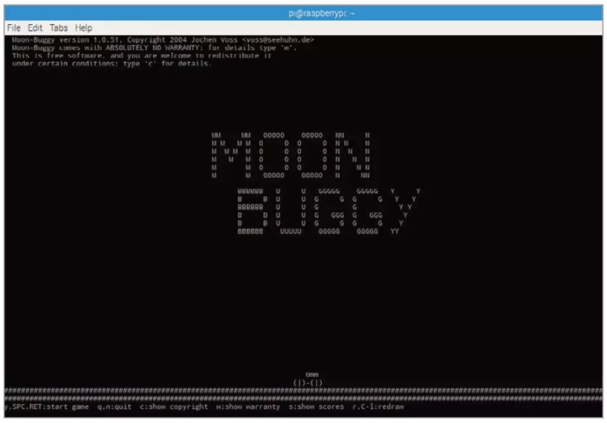
Based on the classic 1982 arcade game, Moon Patrol, Moon Buggy appeared on home computers in 1985 amid much praise. It's a cracking Atari game available in the Linux Terminal by entering:

```
sudo apt-get install moon-buggy
```

Then:

```
moon-buggy
```

Enjoy.

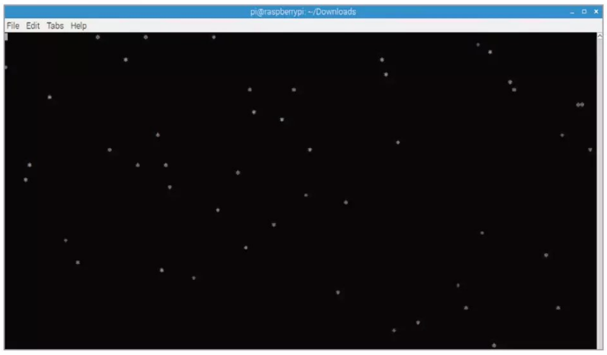


```
Moon-Buggy version 1.0.51, Copyright 2004 Jochen Voss <voss@seehuhn.de>
Moon-Buggy comes with ABSOLUTELY NO WARRANTY; for details type 'w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type 'c' for details.

y,SPC,RET:start game  q,n:quit  c:show copyright  w:show warranty  s:show scores  r,C-l:redraw
```

LET IT SNOW

Snowing in the Terminal console isn't something you come across every day. If you're interested, however, enter:

```
wget

https://gist.githubusercontent.com/sontek/1505483/raw/7d024716ea57e69fb52632fee09f42753361c4a2/snowjob.sh

chmod +x snowjob.sh

./snowjob.sh
```

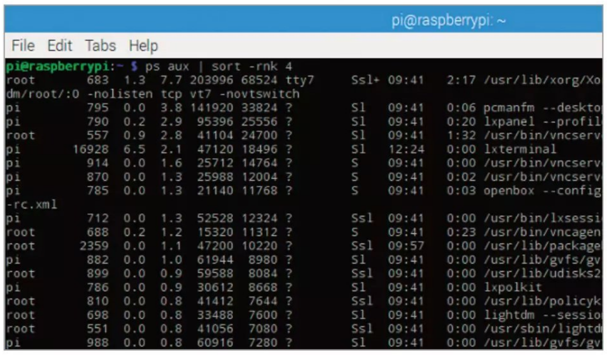


MEMORY HOGS

If you need to see which apps are consuming the most memory on your Raspberry Pi, simply enter:

```
ps aux | sort -rnk 4
```

This sorts the output by system memory use.



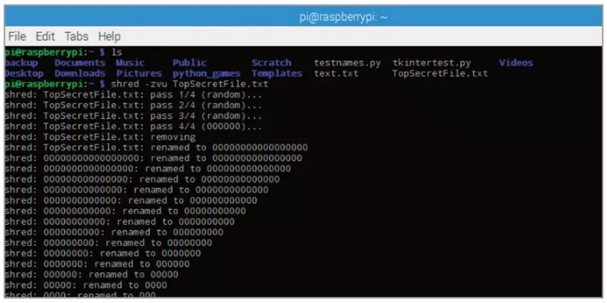
```
pi@raspberrypi:~ $ ps aux | sort -rnk 4
root       683  1.3  7.7 203996 68524 tty7     Ssl+ 09:41   2:17 /usr/lib/xorg/Xo
dm/root/:0 -nolisten tcp vt7 -novtswitch
pi         795  0.0  3.8 141920 33824 ?        Sl   09:41   0:06 pcmanfm --deskto
pi         790  0.2  2.9  95396 25556 ?        Sl   09:41   0:20 lxpanel --profil
root       557  0.9  2.8  41104 24700 ?        Sl   09:41   1:32 /usr/bin/vncserv
pi       16928  6.5  2.1  47120 18496 ?        Sl   12:24   0:00 lxterminal
pi         914  0.0  1.6  25712 14764 ?        S    09:41   0:00 /usr/bin/vncserv
pi         870  0.0  1.3  25988 12004 ?        S    09:41   0:02 /usr/bin/vncserv
pi         785  0.0  1.3  21140 11768 ?        S    09:41   0:03 openbox --config
-rc.xml
pi         712  0.0  1.3  52528 12324 ?        Ssl  09:41   0:00 /usr/bin/lxsessi
root       688  0.2  1.2  15320 11312 ?        S    09:41   0:23 /usr/bin/vncagen
root      2359  0.0  1.1  47200 10220 ?        Ssl  09:57   0:00 /usr/lib/package
pi         882  0.0  1.0  61944  8980 ?        Sl   09:41   0:00 /usr/lib/gvfs/gv
root       899  0.0  0.9  59588  8084 ?        Ssl  09:41   0:00 /usr/lib/udisks2
pi         786  0.0  0.9  30612  8668 ?        Sl   09:41   0:00 lxpolkit
root       810  0.0  0.8  41412  7644 ?        Ssl  09:41   0:00 /usr/lib/policyk
root       698  0.0  0.8  33488  7600 ?        Sl   09:41   0:00 lightdm --sessio
root       551  0.0  0.8  41056  7080 ?        Ssl  09:41   0:00 /usr/sbin/lightd
pi         988  0.0  0.8  60916  7280 ?        Sl   09:41   0:00 /usr/lib/gvfs/gv
```

SHREDDER

When you delete a file, there's still a chance of someone with the right software being able to retrieve it. However, files can be securely and permanently deleted using Shred:

```
shred -zvu NAMEOFFILE.txt
```

Replace NAMEOFFILE with the name of the file to delete.



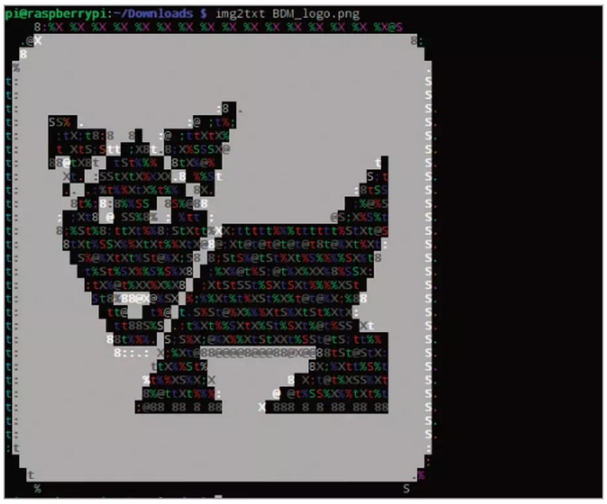
```
pi@raspberrypi:~ $ ls
backup   Documents  Music     Public        Scratch    testnames.py  tkintertest.py    Videos
Desktop  Downloads  Pictures  python_games  Templates  text.txt      TopSecretFile.txt
pi@raspberrypi:~ $ shred -zvu TopSecretFile.txt
shred: TopSecretFile.txt: pass 1/4 (random)...
shred: TopSecretFile.txt: pass 2/4 (random)...
shred: TopSecretFile.txt: pass 3/4 (random)...
shred: TopSecretFile.txt: pass 4/4 (000000)...
shred: TopSecretFile.txt: removing
shred: TopSecretFile.txt: renamed to 00000000000000000
shred: 00000000000000000: renamed to 0000000000000000
shred: 0000000000000000: renamed to 000000000000000
shred: 000000000000000: renamed to 00000000000000
shred: 00000000000000: renamed to 0000000000000
shred: 0000000000000: renamed to 000000000000
shred: 000000000000: renamed to 00000000000
shred: 00000000000: renamed to 0000000000
shred: 0000000000: renamed to 000000000
shred: 000000000: renamed to 00000000
shred: 00000000: renamed to 0000000
shred: 0000000: renamed to 000000
shred: 000000: renamed to 00000
shred: 00000: renamed to 0000
shred: 0000: renamed to 000
```

ASCII ART

ASCII art can be quite striking when applied to some images. However, it's often difficult to get just right. You can create some great ASCII art from the images you already have on the Raspberry Pi by using img2txt:

```
img2txt NAMEOFIMAGEFILE.png
```

Replace NAMEOFIMAGEFILE with the actual name of the image file on your system.



```
pi@raspberrypi:~/Downloads $ img2txt BDM_logo.png
```

BBS

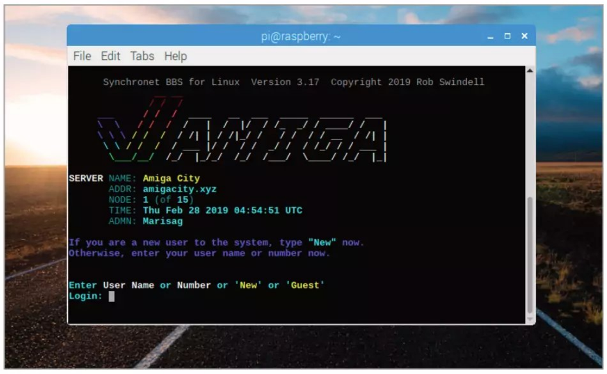
Back in the days of dial-up connections, the online world was made up of Bulletin Board Systems. These remote servers provided hangouts for users to chat, swap code, play games and more. Using telnet in Linux, we can still connect to some active BBSes. First, install Telnet with:

```
sudo apt-get install telnet
```

Then:

```
telnet amigacity.xyz
```

You will be connected to a BBS dedicated to the Commodore Amiga. There's plenty more, which you can find at www.telnetbbsguide.com.



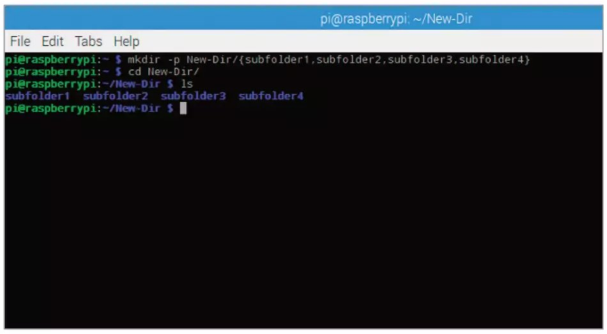
```
     Synchronet BBS for Linux  Version 3.17  Copyright 2019 Rob Swindell

SERVER NAME: Amiga City
       ADDR: amigacity.xyz
       NODE: 1 (of 15)
       TIME: Thu Feb 28 2019 04:54:51 UTC
       ADMN: Marisag

If you are a new user to the system, type "New" now.
Otherwise, enter your user name or number now.


Enter User Name or Number or 'New' or 'Guest'
Login:
```

DIRECTORY TREES

If you want to create an entire directory (or folder) tree with a single command, you can use:

```
mkdir -p New-Dir/{subfolder1,subfolder2,subfolder3,s
ubfolder4}
```

This creates a New-Dir with four sub folders within.

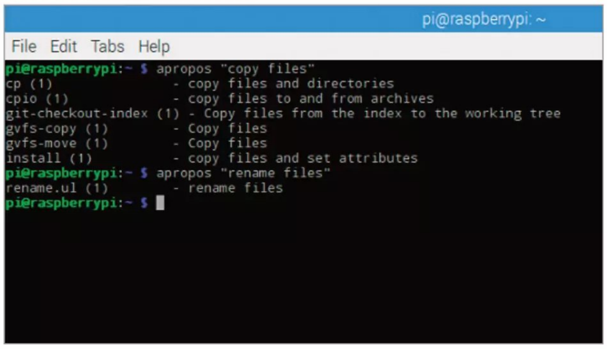


```
pi@raspberrypi:~ $ mkdir -p New-Dir/{subfolder1,subfolder2,subfolder3,subfolder4}
pi@raspberrypi:~ $ cd New-Dir/
pi@raspberrypi:~/New-Dir $ ls
subfolder1  subfolder2  subfolder3  subfolder4
pi@raspberrypi:~/New-Dir $
```

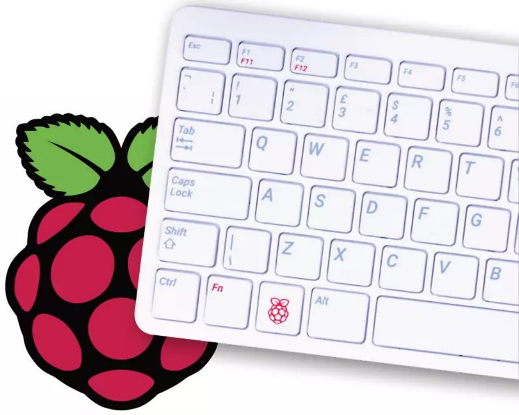
FORGOTTEN COMMANDS

It's not easy trying to remember all the available Linux commands. Thankfully, you can use apropos to help. Simply use it with a description of the command:

```
apropos "copy files"
apropos "rename files"
```



```
pi@raspberrypi:~ $ apropos "copy files"
cp (1)                 - copy files and directories
cpio (1)               - copy files to and from archives
git-checkout-index (1) - Copy files from the index to the working tree
gvfs-copy (1)          - Copy files
gvfs-move (1)          - Copy files
install (1)            - copy files and set attributes
pi@raspberrypi:~ $ apropos "rename files"
rename.ul (1)          - rename files
pi@raspberrypi:~ $
```



Command Line Quick Reference

When you start using Linux full time, you will quickly realise that the graphical interfaces of Ubuntu, Mint, etc. are great for many tasks but not great for all tasks. Understanding how to use the command line not only builds your understanding of Linux but also improves your knowledge of coding and programming in general. Our command line quick reference guide is designed to help you master Linux quicker.

TOP 10 COMMANDS

These may not be the most common commands used by everyone but they will certainly feature frequently for many users of Linux and the command line.

cd

The 'cd' command is one of the commands you will use the most at the command line in Linux. It allows you to change your working directory. You use it to move around within the hierarchy of your file system. You can also use chdir.

ls

The 'ls' command shows you the files in your current directory. Used with certain options, it lets you see file sizes, when files where created and file permissions. For example, `ls ~` shows you the files that are in your home directory.

cp

The 'cp' command is used to make copies of files and directories. For example, `cp file sub` makes an exact copy of the file whose name you entered and names the copy sub but the first file will still exist with its original name.

pwd

The 'pwd' command prints the full pathname of the current working directory (pwd stands for "print working directory"). Note that the GNOME terminal also displays this information in the title bar of its window.

clear

The 'clear' command clears your screen if this is possible. It looks in the environment for the terminal type and then in the terminfo database to figure out how to clear the screen. This is equivalent to typing Control-L when using the bash shell.

mv

The 'mv' command moves a file to a different location or renames a file. For example `mv file sub` renames the original file to sub. `mv sub ~/Desktop` moves the file 'sub' to your desktop directory but does not rename it. You must specify a new filename to rename a file.

chown

The 'chown' command changes the user and/or group ownership of each given file. If only an owner (a user name or numeric user ID) is given, that user is made the owner of each given file, and the files' group is not changed.

chmod

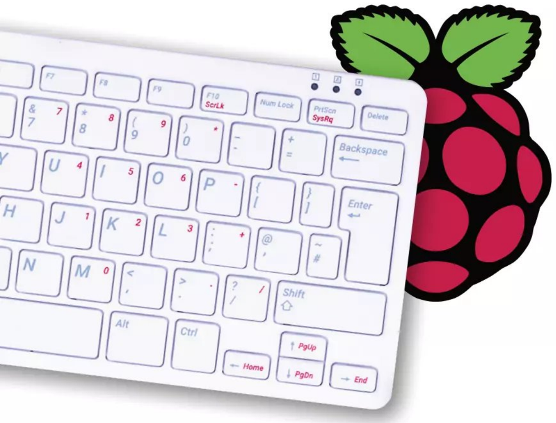
The 'chmod' command changes the permissions on the files listed. Permissions are based on a fairly simple model. You can set permissions for user, group and world and you can set whether each can read, write and or execute the file.

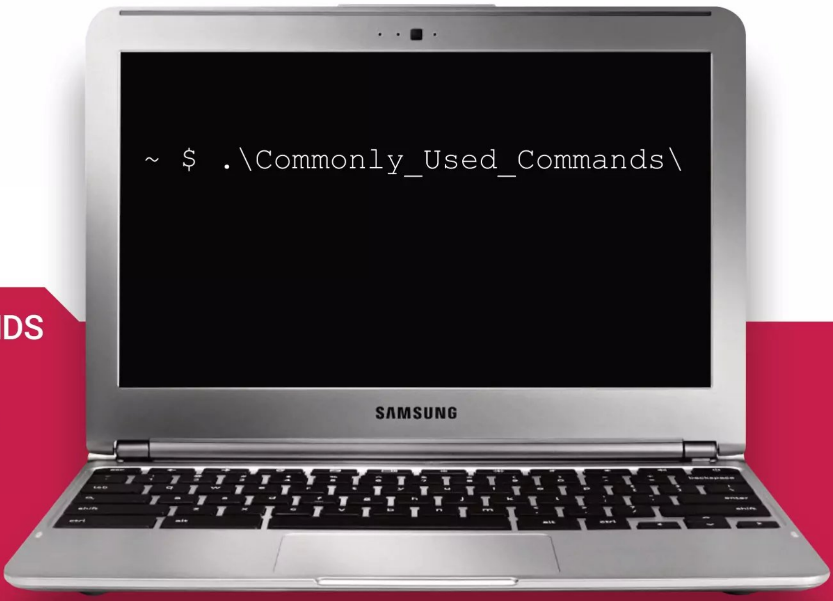
rm

The 'rm' command removes (deletes) files or directories. The removal process unlinks a filename in a filesystem from data on the storage device and marks that space as usable by future writes. In other words, removing files increases the amount of available space on your disk.

mkdir

Short for "make directory", 'mkdir' is used to create directories on a file system, if the specified directory does not already exist. For example, `mkdir work` creates a work directory. More than one directory may be specified when calling mkdir.





```
~ $ .\Commonly_Used_Commands\
```

USEFUL HELP/INFO COMMANDS

The following commands are useful for when you are trying to learn more about the system or program you are working with in Linux. You might not need them every day, but when you do, they will be invaluable.

free The 'free' command displays the total amount of free and used physical and swap memory in the system. For example, `free -m` gives the information using megabytes.

df The 'df' command displays filesystem disk space usage for all partitions. The command `df-h` is probably the most useful (the -h means human-readable).

top The 'top' program provides a dynamic real-time view of a running system. It can display system summary information, as well as a list of processes.

uname The 'uname' command with the `-a` option prints all system information, including machine name, kernel name, version and a few other details.

ps The 'ps' command allows you to view all the processes running on the machine. Every operating system's version of ps is slightly different but all do the same thing.

grep The 'grep' command allows you to search inside a number of files for a particular search pattern and then print matching lines. An example would be: `grep blah file`.

sed The 'sed' command opens a stream editor. A stream editor is used to perform text transformations on an input stream: a file or input from a pipeline.

adduser The 'adduser 'command adds a new user to the system. Similarly, the `addgroup` command adds a new group to the system.

deluser The 'deluser' command removes a user from the system. To remove the user's files and home directory, you need to add the `-remove-home` option.

delgroup The 'delgroup' command removes a group from the system. You cannot remove a group that is the primary group of any users.

man man The 'man man' command brings up the manual entry for the man command, which is a great place to start when using it.

man intro The 'man intro' command is especially useful. It displays the Introduction to User Commands, which is a well written, fairly brief introduction to the Linux command line.

A-Z of Linux Commands

There are literally thousands of Linux commands, so while this is not a complete A-Z, it does contain many of the commands you will most likely need. You will probably find that you end up using a smaller set of commands over and over again but having an overall knowledge is still very useful.

A

`adduser`	Add a new user
`arch`	Print machine architecture
`awk`	Find and replace text within file(s)

B

`bc`	An arbitrary precision calculator language

C

`cat`	Concatenate files and print on the standard output
`chdir`	Change working directory
`chgrp`	Change the group ownership of files
`chroot`	Change root directory
`cksum`	Print CRC checksum and byte counts
`cmp`	Compare two files
`comm`	Compare two sorted files line by line
`cp`	Copy one or more files to another location
`crontab`	Schedule a command to run at a later time
`csplit`	Split a file into context-determined pieces
`cut`	Divide a file into several parts

D

`date`	Display or change the date & time
`dc`	Desk calculator
`dd`	Data Dump, convert and copy a file
`diff`	Display the differences between two files
`dirname`	Convert a full path name to just a path
`du`	Estimate file space usage

E

`echo`	Display message on screen
`ed`	A line oriented text editor (edlin)
`egrep`	Search file(s) for lines that match an extended expression
`env`	Display, set or remove environment variables
`expand`	Convert tabs to spaces
`expr`	Evaluate expressions

F

`factor`	Print prime factors
`fdisk`	Partition table manipulator for Linux
`fgrep`	Search file(s) for lines that match a fixed string
`find`	Search for files that meet a desired criteria
`fmt`	Reformat paragraph text
`fold`	Wrap text to fit a specified width
`format`	Format disks or tapes
`fsck`	Filesystem consistency check and repair

G

`gawk`	Find and Replace text within file(s)
`grep`	Search file(s) for lines that match a given pattern
`groups`	Print group names a user is in
`gzip`	Compress or decompress named file(s)

H

`head`	Output the first part of file(s)
`hostname`	Print or set system name

I

`id`	Print user and group ids
`info`	Help info
`install`	Copy files and set attributes

J

`join`	Join lines on a common field

K

`kill`	Stop a process from running

L

`less`	Display output one screen at a time
`ln`	Make links between files
`locate`	Find files
`logname`	Print current login name
`lpc`	Line printer control program
`lpr`	Off line print
`lprm`	Remove jobs from the print queue

M

`man`	See Help manual
`mkdir`	Create new folder(s)
`mkfifo`	Make FIFOs (named pipes)
`mknod`	Make block or character special files
`more`	Display output one screen at a time
`mount`	Mount a file system

N

`nice`	Set the priority of a command or job
`nl`	Number lines and write files
`nohup`	Run a command immune to hangups

P

`passwd`	Modify a user password
`paste`	Merge lines of files
`pathchk`	Check file name portability
`pr`	Convert text files for printing
`printcap`	Printer capability database
`printenv`	Print environment variables
`printf`	Format and print data

Q

`quota`	Display disk usage and limits
`quotacheck`	Scan a file system for disk usage
`quotactl`	Set disk quotas

R

`ram`	Ram disk device
`rcp`	Copy files between two machines
`rm`	Remove files
`rmdir`	Remove folder(s)
`rpm`	Remote Package Manager
`rsync`	Remote file copy (synchronise file trees)

S

`screen`	Terminal window manager
`sdiff`	Merge two files interactively
`select`	Accept keyboard input
`seq`	Print numeric sequences
`shutdown`	Shutdown or restart Linux
`sleep`	Delay for a specified time
`sort`	Sort text files
`split`	Split a file into fixed-size pieces
`su`	Substitute user identity
`sum`	Print a checksum for a file
`symlink`	Make a new name for a file
`sync`	Synchronise data on disk with memory

T

`tac`	Concatenate and write files in reverse
`tail`	Output the last part of files
`tar`	Tape Archiver
`tee`	Redirect output to multiple files
`test`	Evaluate a conditional expression
`time`	Measure Program Resource Use
`touch`	Change file timestamps
`top`	List processes running on the system
`traceroute`	Trace Route to Host
`tr`	Translate, squeeze and or delete characters
`tsort`	Topological sort

U

`umount`	Unmount a device
`unexpand`	Convert spaces to tabs
`uniq`	Uniquify files
`units`	Convert units from one scale to another
`unshar`	Unpack shell archive scripts
`useradd`	Create new user account
`usermod`	Modify user account
`users`	List users currently logged in

V

`vdir`	Verbosely list directory contents (`ls -l -b')

W

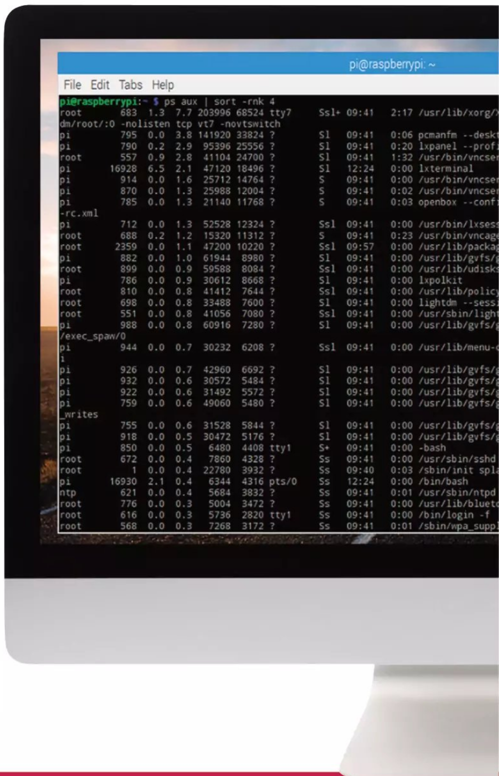
`watch`	Execute or display a program periodically
`wc`	Print byte, word, and line counts
`whereis`	Report all known instances of a command
`which`	Locate a program file in the user's path
`who`	Print all usernames currently logged in
`whoami`	Print the current user id and name

X

`xargs`	Execute utility, passing constructed argument list(s)

Y

`yes`	Print a string until interrupted



```
pi@raspberrypi:~ $ ps aux | sort -rnk 4
root       683  1.3  7.7 203996 68524 tty7     Ssl+ 09:41   2:17 /usr/lib/xorg/
dm/root/:0 -nolisten tcp vt7 -novtswitch
pi         795  0.0  3.8 141920 33824 ?        Sl   09:41   0:06 pcmanfm --desk
pi         790  0.2  2.9  95396 25556 ?        Sl   09:41   0:20 lxpanel --prof
root       557  0.9  2.8  41104 24700 ?        Sl   09:41   1:32 /usr/bin/vncse
pi       16928  6.5  2.1  47120 18496 ?        Sl   12:24   0:00 lxterminal
pi         914  0.0  1.6  25712 14764 ?        S    09:41   0:00 /usr/bin/vncse
pi         870  0.0  1.3  25988 12004 ?        S    09:41   0:02 /usr/bin/vncse
pi         785  0.0  1.3  21140 11768 ?        S    09:41   0:03 openbox --conf
-rc.xml
pi         712  0.0  1.3  52528 12324 ?        Ssl  09:41   0:00 /usr/bin/lxses
root       688  0.2  1.2  15320 11312 ?        S    09:41   0:23 /usr/bin/vncag
root      2359  0.0  1.1  47200 10220 ?        Ssl  09:57   0:00 /usr/lib/packa
pi         882  0.0  1.0  61944  8980 ?        Sl   09:41   0:00 /usr/lib/gvfs/
root       899  0.0  0.9  59588  8084 ?        Ssl  09:41   0:00 /usr/lib/udisk
pi         786  0.0  0.9  30612  8668 ?        Sl   09:41   0:00 lxpolkit
root       810  0.0  0.8  41412  7644 ?        Ssl  09:41   0:00 /usr/lib/polic
root       698  0.0  0.8  33488  7600 ?        Sl   09:41   0:00 lightdm --sess
root       551  0.0  0.8  41056  7080 ?        Ssl  09:41   0:00 /usr/sbin/ligh
pi         988  0.0  0.8  60916  7280 ?        Sl   09:41   0:00 /usr/lib/gvfs/
/exec_spaw/0
pi         944  0.0  0.7  30232  6208 ?        Ssl  09:41   0:00 /usr/lib/menu-
i
pi         926  0.0  0.7  42960  6692 ?        Sl   09:41   0:00 /usr/lib/gvfs/
pi         932  0.0  0.6  30572  5484 ?        Sl   09:41   0:00 /usr/lib/gvfs/
pi         922  0.0  0.6  31492  5572 ?        Sl   09:41   0:00 /usr/lib/gvfs/
pi         759  0.0  0.6  49060  5480 ?        Sl   09:41   0:00 /usr/lib/gvfs/
_writes
pi         755  0.0  0.6  31528  5844 ?        Sl   09:41   0:00 /usr/lib/gvfs/
pi         918  0.0  0.5  30472  5176 ?        Sl   09:41   0:00 /usr/lib/gvfs/
pi         850  0.0  0.5   6480  4408 tty1     S+   09:41   0:00 -bash
root       672  0.0  0.4   7860  4328 ?        Ss   09:41   0:00 /usr/sbin/sshd
root         1  0.0  0.4  22780  3932 ?        Ss   09:40   0:03 /sbin/init spl
pi       16930  2.1  0.4   6344  4316 pts/0    Ss   12:24   0:00 /bin/bash
ntp        621  0.0  0.4   5684  3832 ?        Ss   09:41   0:01 /usr/sbin/ntpd
root       776  0.0  0.3   5004  3472 ?        Ss   09:41   0:00 /usr/lib/bluet
root       616  0.0  0.3   5736  2820 tty1     Ss   09:41   0:00 /bin/login -f
root       568  0.0  0.3   7268  3172 ?        Ss   09:41   0:01 /sbin/wpa_supp
```

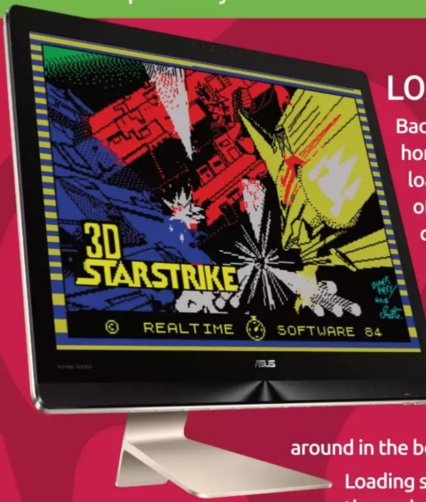
Pi Projects

The Raspberry Pi is all about projects. We've seen the Pi attached to a satellite taking high-resolution images of the Earth, as well as by scientists monitoring habitats in some of the most extreme places on the planet. And enthusiasts the world over have used a Raspberry Pi as the driving force of their unique projects.

Want to use the Pi to see where the International Space Station is? How about creating a retro gaming system, or a media centre, or even connecting to a Bulletin Board System? You'll find these projects and more in this section. Where next is up to you. Let your imagination fly with the Raspberry Pi.

Creating a Loading Screen

If you're looking to add a little something extra to your Python code, then consider including a loading screen. The loading screen is a short introduction, or piece of art, that appears before the main part of your code.



LOAD""

Back in the 80s, in the 8-bit home computing era, loading screens were often used to display the cover of a game as it loaded from tape. The image would load itself in, usually one-line-at-a-time, then proceed to colour itself in while the loading raster bars danced around in the borders of the screen.

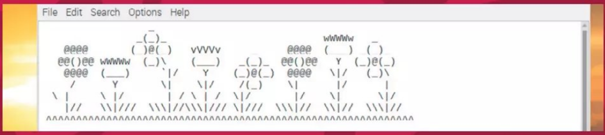
Loading screens were a part of the package and often the buy-in for the whole game as an experience. Some loading screens featured animations, or a countdown for time remaining as the game loads, while others even went so far as to include some kind of playable game. The point being: a loading screen is not just an artistic part of computing history, but an introduction to the program that's about to be run.

While these days loading screens may no longer be with us, in terms of modern gaming we can still include them in our own Python content. Either just for fun, or to add a little retro-themed spice to the mix.

SCREEN$

Creating a loading screen in Python is remarkably easy. You have several options to hand: you can create a tkinter window and display an image, followed by a brief pause, before starting your main code, or you could opt for a console-based ASCII art version that loads one-line-at-a-time. Let's look at the latter and see how it works.

First you'll need some ASCII art, you can look up plenty of examples online, or use an image to ASCII Art converter to transform any images you have to ASCII. When you have your ASCII art, drop it into a newly created folder inside a normal text file.



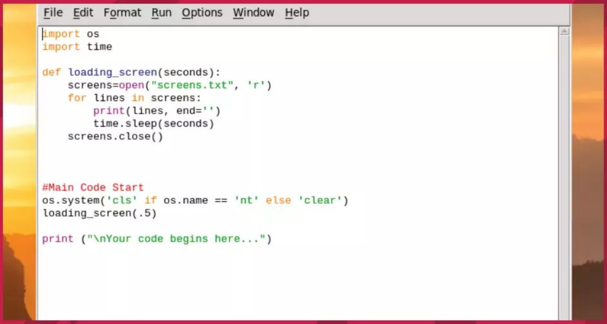
```
                 _
              _(_)_                          wWWWw   _
   @@@@       ( )@( )   vVVVv          @@@@  (___)  ( )
  @@()@@ wWWWw  (_)\    (___)  _(_)_  @@()@@   Y  (_)@(_)
   @@@@  (___)     `|/    Y    (_)@(_)  @@@@   \|/   (_)\
    /      Y       \|    \|/    /(_)    \|      |/      |
 \ |     \ |/       | / \ | / \|/       |/    \|      \|/
   |//   \\|///  \\\|//\\\|/// \|///  \\\|//  \\|//  \\\|//
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

Save the file, call it screens.txt for now.

THE CODE

Launch Python and enter the following code to a New File:

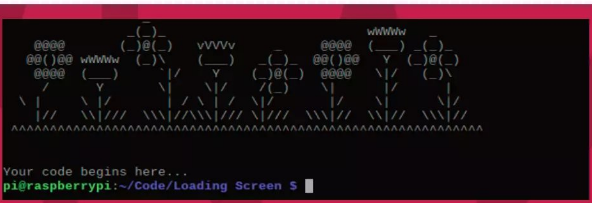
```
import os
import time

def loading_screen(seconds):
    screens=open("screens.txt", 'r')
    for lines in screens:
        print(lines, end='')
        time.sleep(seconds)
    screens.close()

#Main Code Start
os.system('cls' if os.name == 'nt' else 'clear')
loading_screen(.5)

print ("\nYour code begins here...")
```



```
import os
import time

def loading_screen(seconds):
    screens=open("screens.txt", 'r')
    for lines in screens:
        print(lines, end='')
        time.sleep(seconds)
    screens.close()


#Main Code Start
os.system('cls' if os.name == 'nt' else 'clear')
loading_screen(.5)

print ("\nYour code begins here...")
```

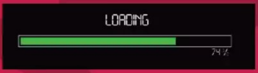
The code is quite simple: import the OS and Time modules and then create a Python function called loading_screen with a (seconds) option. Within the function, open the text file with the ASCII art as read-only and create a `For` loop that'll read the text file one-line-at-a-time. Next, print the lines – incidentally, the `lines, end=''` element will strip the newline from the text document, without it you'll end up with a double-line spaced display. Include the timing in seconds and close the text file buffer.

The final part of the code, `#Main Code Start`, is where you'll clear the screen (CLS for Windows, Clear for Linux (Raspberry Pi) and macOS) and call the function, together with the output number of seconds to wait for each line to be written to the screen – in this case, .5 of a second.



```
Your code begins here...
pi@raspberrypi:~/Code/Loading Screen $
```

Save the code as screens.py, drop into a Command Prompt or Terminal and execute it. The screen will clear and your ASCII art inside the text file will load in line-by-line, creating a loading screen effect.



LOADING...

Another favourite introduction screen is that of a simple loading animation, where the word loading is displayed, followed by some characters, and a percentage of the program loaded. While it may not take long for your Python code to load, the effect can be interesting.

Create a New File in Python and enter the following code:

```
import os
import time

def loading_bar(seconds):
  for loading in range(0,seconds+1):
    percent = (loading * 100) // seconds
    print("\n")
    print("Loading...")
    print("<" + ("-" * loading) + (" " *
(seconds-loading)) + "> " + str(percent) + "%")
    print("\n")
    time.sleep(1)
    os.system('cls' if os.name == 'nt' else
'clear')

#Main Code Start
loading_bar(10)

print ("\nYour code begins here...")
```

The code works in much the same way as the previous, except, instead of reading from a text file, it's running through a For loop that prints Loading... followed by an animation of sorts, along with a percentage counter; clearing the screen every second and displaying the new results. It's simple, yes, but quite effective in its design.

COMBINING THE TWO

How about combining the two elements we've looked at? Let's begin with a Loading... progress bar, followed by the loading screen. After that, you can include your own code and continue your program. Here's the code:

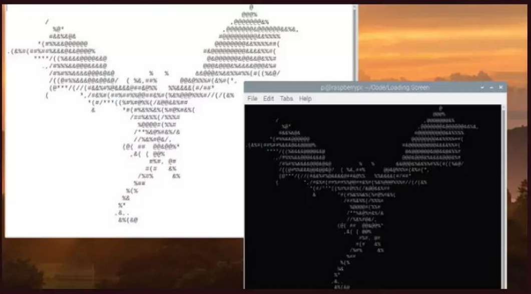
```
import os
import time

def loading_bar(seconds):
  for loading in range(0,seconds+1):
    percent = (loading * 100) // seconds
    print("\n")
    print("Loading...")
    print("<" + ("-" * loading) + (" " *
(seconds-loading)) + "> " + str(percent) + "%")
    print("\n")
    time.sleep(1)
    os.system('cls' if os.name == 'nt' else
'clear')

def loading_screen(seconds):
    screens=open("screens.txt", 'r')
    for lines in screens:
        print(lines, end='')
        time.sleep(seconds)
    screens.close()

#Main Code Start
loading_bar(10)
os.system('cls' if os.name == 'nt' else 'clear')
loading_screen(.5)

print ("\nYour code begins here...")
```

You can, of course, pull those functions up wherever and whenever you want in your code and they'll display, as they should, at the beginning. Remember to have the ASCII text file in the same folder as the Python code, or, at the `screens=open("screens.txt", 'r')` part of the code, point to where the text file is located.

ADVENTURE TIME

A good example of using loading screen, ASCII art text images is when coding a text adventure. Once you've established your story, created the characters, events and so on, you could easily incorporate some excellently designed ASCII art to your game.

Imagine coming across a dragon, in game, and displaying its representation as ASCII. You can then load up the image lines, one-by-one, and continue with the rest of the adventure code. It's certainly worth having a play around with and it'll definitely add a little something else extra.



Tracking the ISS with Python

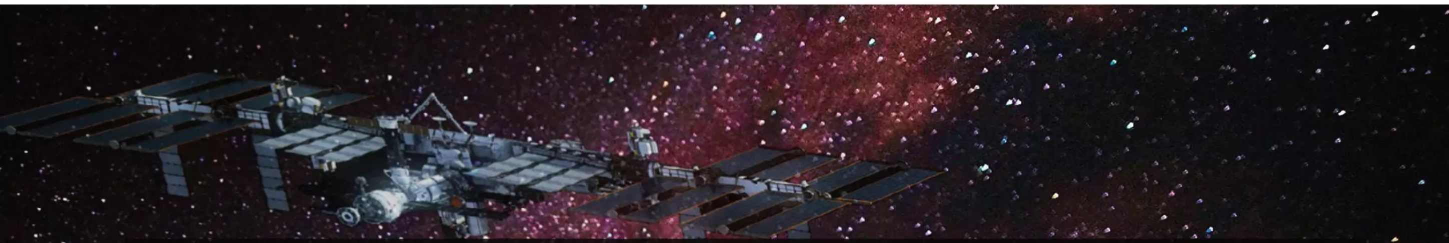
Of the many, amazing human achievements over the past decade or so, the International Space Station tops the bill. This incredible collaboration between nations sees vital experiments carried out in space, as well as observations of our own planet.

TO BOLDLY GO...

Indeed, the ISS is something most of us consider as a worthy example of what can happen when we work together. NASA, among other agencies, uses a wealth of Python code onboard the ISS to help automate routines, alongside its function as an in-between link to translate code from one language to another, and then into a human-readable format. If you're considering a career in space, then learning Python is a must-have skill.

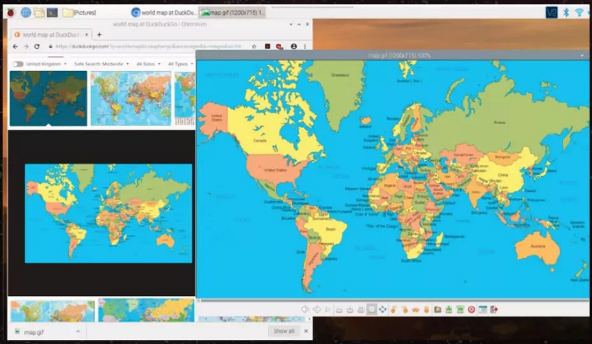
While we're not able to fill you in on all the details regarding the code the ISS utilises, we can look at a fun Python script that will track the ISS, display the number of astronauts on board, and update the station's current latitude and longitude every five seconds, while also displaying your current latitude and longitude.

Displaying all that information in a single screen can become a little cluttered, so in this instance we're going to look at spreading all those details over three screens: a text document window -displaying the astronauts and your current latitude and longitude, a command line (or Terminal window) - displaying the continually updating latitude and longitude of the ISS as it orbits Earth, and a final, large window displaying a map of the world, together with an icon representing the ISS, that's updated as it orbits. Interested? Read on.



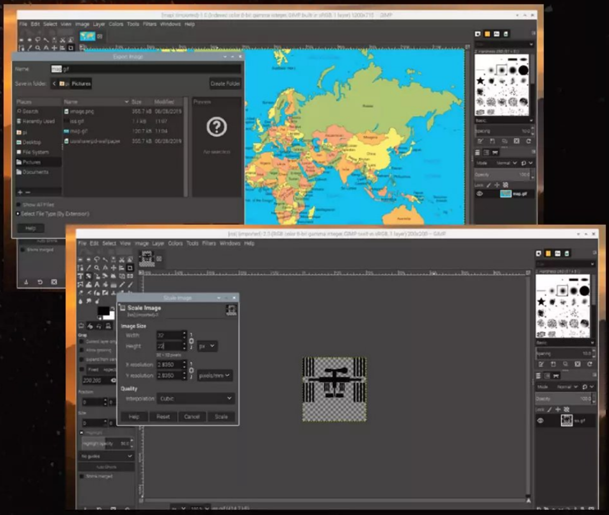
THE GRAPHICS

Firstly, we need to get hold of a map of the world and an image of the ISS, to use as an icon that will be updated according to the position of the space station as it travels over the surface. A quick Google of World Map will help you out here. Look for one that's reasonably large, the one we used for this example was 1280 x 700, and one that has the names of the countries – if you're using this with young people, to help with putting shapes of countries to names.



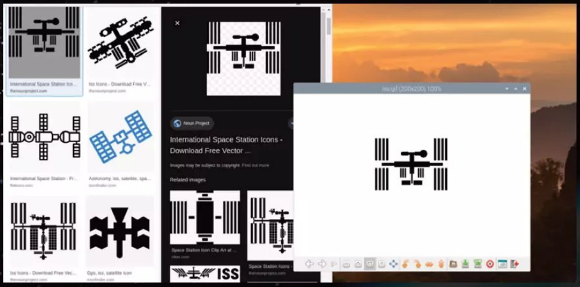
Next, look for an ISS icon. As this is going to be a graphical representation of the location of the ISS, we need the image to be reasonably small so it doesn't drown out the locations on the map, but also prominent enough to see when the map is loaded. We opted for an image that's 32 x 22 pixels in size. Don't worry too much if you're not able to find one that small, you can always resize it in an image-editing app such as GIMP.

As we're going to be using Turtle, a component of tkinter, the downloaded images will need to be converted to GIF, since this is the default and recommended image format. You can easily look up a converter online, but using GIMP, which is cross-platform and therefore works on both the Raspberry Pi and a Windows PC, will suffice. Simply load the image up in your image editor app and choose Save As, call them map and iss respectively, and click GIF as the image format. Remember to also resize the ISS image before saving it as a GIF.

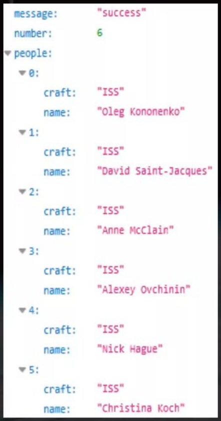


THE CODE

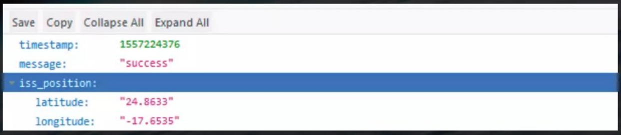
The code we're using here utilises an open source API (Application Programming Interface) to retrieve real-time data online regarding the status of the ISS. An API enables applications to communicate with one another by providing the raw data that a programmer can pull out and interact with in their own code. In this case, the API in question is a web-based collection of raw data that's stored in a JSON (JavaScript Object Notation) format – an accessible and easy to use data-interchange interface.



In order for Python to interact with the JSON provided data, it needs to use the urllib.request and json modules. We're also going to be using a new module called geocoder, which will pull up a users' current latitude and longitude based on their IP address. The two JSON APIs can be located at: **http://api.open-notify.org/astros.json**, and, **http://api.open-notify.org/iss-now.json**. One of which contains the data regarding the astronauts onboard the ISS, and the other contains the data regarding the current location of the ISS in longitude and latitude.



```
message:     "success"
number:      6
▼ people:
  ▼ 0:
      craft: "ISS"
      name:  "Oleg Kononenko"
  ▼ 1:
      craft: "ISS"
      name:  "David Saint-Jacques"
  ▼ 2:
      craft: "ISS"
      name:  "Anne McClain"
  ▼ 3:
      craft: "ISS"
      name:  "Alexey Ovchinin"
  ▼ 4:
      craft: "ISS"
      name:  "Nick Hague"
  ▼ 5:
      craft: "ISS"
      name:  "Christina Koch"
```



```
timestamp:     1557224376
message:       "success"
▼ iss_position:
    latitude:  "24.8633"
    longitude: "-17.6535"
```

Let's begin by breaking the code into bite-sized chunks:

```
import json, turtle, urllib.request, time, webbrowser
import geocoder # need to pip install geocoder for your lat/long to work.
```

First, we need to import the modules used in the code: json, turtle, urllib.request, time, and webbrowser. The json and urlib.request modules deal with the data being pulled from the APIs, turtle will display the graphics, and time you already know. The Webbrowser module will open text files in the default browser or default text reader application. The geocoder module is a new element and you will need to install it with: `pip install geocoder` (or try `pip3 install geocoder`, if the pip install command still won't load the module). Geocoder can retrieve a users' location based on their IP address, as each ISP will have a geo-specific IP.

```
#Retrieve the names of all the astronauts currently on board the ISS, and own lat/long - write to a file and display
url = "http://api.open-notify.org/astros.json"
response = urllib.request.urlopen(url)
result = json.loads(response.read())
a=open("iss.txt","w")
a.write("There are currently " + str(result["number"]) + " astronauts on the ISS:\n\n")

people = result["people"]

for p in people:
  a.write(p["name"] + " - on board" + "\n")

g=geocoder.ip('me') # need to pip install geocoder, and import as in the headers above
a.write("\nYour current Lat/Long is: " + str(g.latlng)) # prints your current lat/long in the text file.
a.close()
webbrowser.open("iss.txt")
```

This section will use the json and urllib.request modules to pull the data from the API that contains the names of the astronauts onboard the ISS. It then creates a new text file called iss.txt, where it'll write 'There are currently X astronauts on the ISS...' and list them by name. The second part of this section will then use the geocoder module to retrieve your current latitude and longitudes, based on your IP address, and also write that information to the end of the text file that contains the names of the astronauts. The last line, webbrowser.open("iss.txt") will use the webbroser module to open and display the newly written text file in either the system's default text file reading app or the default web browser; either will work just fine.

```
#Setup world map in Turtle
screen = turtle.Screen()
screen.setup(1280, 720)
screen.setworldcoordinates(-180, -90, 180, 90)
#Load the world map image
screen.bgpic("map.gif")

screen.register_shape("iss.gif")
iss = turtle.Turtle()
iss.shape("iss.gif")
iss.setheading(45)
iss.penup()
```

This section of the code sets up the graphical window containing the world map and the ISS icon. To begin, we set up the turtle screen, using the resolution of the world map image we downloaded at the start of this tutorial (1280 x 720). The screen.setworldcoordinates syntax will mark the boundaries of the screen, creating the x and y coordinates of the four corners of the canvas, so that the ISS icon can freely travel across the map of the world and wrap itself back to the opposite side when it reaches an edge. The ISS icon is set as the turtle pen shape, giving it an angle of 45 degrees when moving.

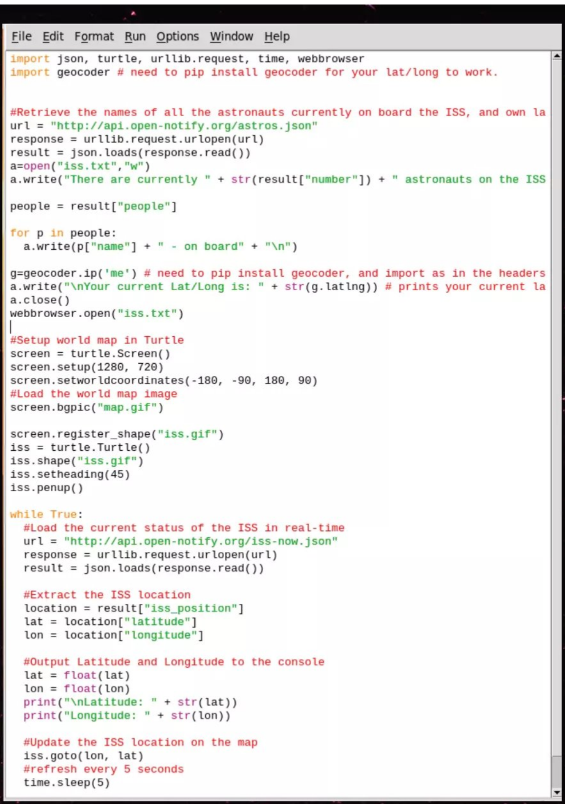
```
while True:
  #Load the current status of the ISS in real-time
  url = "http://api.open-notify.org/iss-now.json"
  response = urllib.request.urlopen(url)
  result = json.loads(response.read())

  #Extract the ISS location
  location = result["iss_position"]
  lat = location["latitude"]
  lon = location["longitude"]

  #Output Latitude and Longitude to the console
  lat = float(lat)
  lon = float(lon)
  print("\nLatitude: " + str(lat))
  print("Longitude: " + str(lon))

  #Update the ISS location on the map
  iss.goto(lon, lat)
  #refresh every 5 seconds
  time.sleep(5)
```

Now for the final part of the code: Here we collect the location data from the ISS status API, pulling out the latitude and longitude elements of the JSON file. The code then prints the latitude and longitude data to the console/Terminal, transferring the data from a float to a string in order to print it correctly. The last section will use the latitude and longitude as variables lat and lon, to update the ISS icon on the map every five seconds.

Here's the code in its entirety:

```
import json, turtle, urllib.request, time, webbrowser
import geocoder # need to pip install geocoder for
your lat/long to work.

#Retrieve the names of all the astronauts currently on
board the ISS, and own lat/long - write to a file and
display
url = “http://api.open-notify.org/astros.json”
response = urllib.request.urlopen(url)
result = json.loads(response.read())
a=open(“iss.txt”,”w”)
a.write(“There are currently “ + str(result[“number”])
+ “ astronauts on the ISS:\n\n”)

people = result[“people”]

for p in people:
  a.write(p[“name”] + “ - on board” + “\n”)

g=geocoder.ip(‘me’) # need to pip install geocoder,
and import as in the headers above
a.write(“\nYour current Lat/Long is: “ + str(g.
latlng)) # prints your current lat/long in the text
file.
a.close()
webbrowser.open(“iss.txt”)

#Setup world map in Turtle
screen = turtle.Screen()
screen.setup(1280, 720)
screen.setworldcoordinates(-180, -90, 180, 90)
#Load the world map image
screen.bgpic(“map.gif”)

screen.register_shape(“iss.gif”)
iss = turtle.Turtle()
iss.shape(“iss.gif”)
iss.setheading(45)
iss.penup()

while True:
  #Load the current status of the ISS in real-time
  url = “http://api.open-notify.org/iss-now.json”
  response = urllib.request.urlopen(url)
  result = json.loads(response.read())

  #Extract the ISS location
  location = result[“iss_position”]
  lat = location[“latitude”]
  lon = location[“longitude”]

  #Output Latitude and Longitude to the console
  lat = float(lat)
  lon = float(lon)
  print(“\nLatitude: “ + str(lat))
  print(“Longitude: “ + str(lon))

  #Update the ISS location on the map
  iss.goto(lon, lat)
  #refresh every 5 seconds
  time.sleep(5)
```

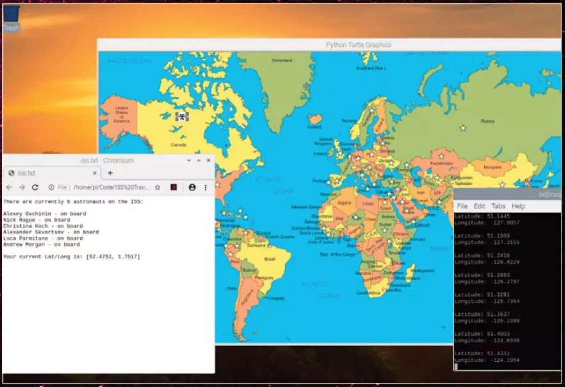
Create a new folder in your system, called ISSTrack (for example), and place the two graphics as well as the Python code itself.



```
import json, turtle, urllib.request, time, webbrowser
import geocoder # need to pip install geocoder for your lat/long to work.


#Retrieve the names of all the astronauts currently on board the ISS, and own la
url = "http://api.open-notify.org/astros.json"
response = urllib.request.urlopen(url)
result = json.loads(response.read())
a=open("iss.txt","w")
a.write("There are currently " + str(result["number"]) + " astronauts on the ISS

people = result["people"]

for p in people:
  a.write(p["name"] + " - on board" + "\n")

g=geocoder.ip('me') # need to pip install geocoder, and import as in the headers
a.write("\nYour current Lat/Long is: " + str(g.latlng)) # prints your current la
a.close()
webbrowser.open("iss.txt")

#Setup world map in Turtle
screen = turtle.Screen()
screen.setup(1280, 720)
screen.setworldcoordinates(-180, -90, 180, 90)
#Load the world map image
screen.bgpic("map.gif")

screen.register_shape("iss.gif")
iss = turtle.Turtle()
iss.shape("iss.gif")
iss.setheading(45)
iss.penup()

while True:
  #Load the current status of the ISS in real-time
  url = "http://api.open-notify.org/iss-now.json"
  response = urllib.request.urlopen(url)
  result = json.loads(response.read())

  #Extract the ISS location
  location = result["iss_position"]
  lat = location["latitude"]
  lon = location["longitude"]

  #Output Latitude and Longitude to the console
  lat = float(lat)
  lon = float(lon)
  print("\nLatitude: " + str(lat))
  print("Longitude: " + str(lon))

  #Update the ISS location on the map
  iss.goto(lon, lat)
  #refresh every 5 seconds
  time.sleep(5)
```

RUNNING THE CODE

The code is best executed from the command line, or Terminal. Clear your desktop, enter your command line, and navigate to where you've saved the code plus the two graphics. Launch the code with either: `python3 ISSTrack.py`, or, `python ISSTrack.py` (depending on whether you're using a Raspberry Pi/Linux, or a Windows PC, and what you've called the Python code).



```
There are currently 6 astronauts on the ISS:

Alexey Ovchinin - on board
Nick Hague - on board
Christina Koch - on board
Alexander Skvortsov - on board
Luca Parmitano - on board
Andrew Morgan - on board

Your current Lat/Long is: [52.4752, 1.7517]
```

```
Latitude: 51.1445
Longitude: -127.9057

Latitude: 51.1989
Longitude: -127.3155

Latitude: 51.2418
Longitude: -126.8228

Latitude: 51.2863
Longitude: -126.2797

Latitude: 51.3281
Longitude: -125.7354

Latitude: 51.3637
Longitude: -125.2398

Latitude: 51.4003
Longitude: -124.6936

Latitude: 51.4311
Longitude: -124.1984
```

The code will launch two extra windows together with the command line window you already have open. One will be the text file, containing the named astronauts, along with your current latitude and longitude and the other will be the world map with the ISS icon located wherever the ISS is currently orbiting. The command line window will start scrolling through the changing latitude and longitude of the ISS.

Text Animations

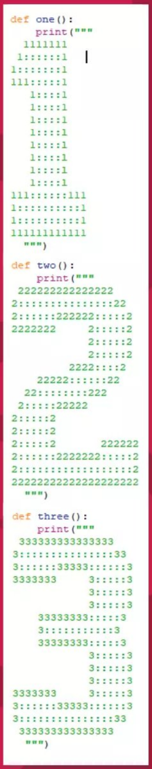
There's a remarkable amount you can do with some simple text and a little Python know-how. Combining what you've already learned, we can create some interesting animation effects from the command line.

THE FINAL COUNTDOWN

Let's begin with some example code that will display a large countdown from ten, then clear the screen and display a message. The code itself is quite simple, but lengthy. You will need to start by importing the OS and Time modules, then start creating functions that display the numbers (see image below) and so on to 10. It'll take some time, but it's worth it in the end. Of course, you can always take a different approach and design the numbers yourself.

The next step of the process is to initialise the code settings and start the countdown:

```
#Initialise settings
start = 10
message = ">       BLAST OFF!!            <"

#Start the countdown
for counter in range(start, 0, -1):
    if counter == 10:
        ten()
    elif counter == 9:
        nine()
    elif counter == 8:
        eight()
    elif counter == 7:
        seven()
    elif counter == 6:
        six()
    elif counter == 5:
        five()
    elif counter == 4:
        four()
    elif counter == 3:
        three()
    elif counter == 2:
        two()
    elif counter == 1:
        one()
    time.sleep(1)
    os.system('cls' if os.name ==
'nt' else 'clear')
```

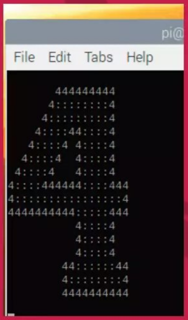


```
def one():
    print("""
  1111111
 1::::::1
1:::::::1
111:::::1
   1::::1
   1::::1
   1::::1
   1::::1
   1::::1
   1::::1
   1::::1
   1::::1
111::::::111
1::::::::::1
1::::::::::1
111111111111
  """)

def two():
    print("""
 222222222222222
2:::::::::::::::22
2::::::222222:::::2
2222222     2:::::2
            2:::::2
            2:::::2
         2222::::2
    22222::::::22
  22::::::::222
 2:::::22222
2:::::2
2:::::2
2:::::2       222222
2::::::2222222:::::2
2::::::::::::::::::2
22222222222222222222
  """)

def three():
    print("""
 333333333333333
3:::::::::::::::33
3::::::33333::::::3
3333333     3:::::3
            3:::::3
            3:::::3
    33333333:::::3
    3:::::::::::3
    33333333:::::3
            3:::::3
            3:::::3
            3:::::3
3333333     3:::::3
3::::::33333::::::3
3:::::::::::::::33
 333333333333333
  """)
```

And finally, we can add a display for the message:

```
#Display the message
print("v^v^v^v^v^v^v^v^v^v^v^v^v^v^v^v")
print("<                              >")
print(message)
print("<                              >")
print("v^v^v^v^v^v^v^v^v^v^v^v^v^v^v^v")
print("\n\n\n")
```

The code in its entirety can be viewed from within our Code Repository: **https://bdmpublications.com/code-portal**, where you're free to copy it to your own Python IDLE and use it as you see fit. The end effect is quite good and it'll be worth adding to your own games, or presentations, in Python.



```
       444444444
      4::::::::4
     4:::::::::4
    4::::44::::4
   4::::4 4::::4
  4::::4  4::::4
 4::::4   4::::4
4::::444444::::444
4::::::::::::::::4
4444444444:::::444
          4::::4
          4::::4
          4::::4
        44::::::44
        4::::::::4
        4444444444
```

To extend the code, or make it easier to use, you can always create the number functions in their own Python code, call it Count.py for example, then import Count at the beginning of a new Python file called Countdown.py, along with the OS and Time modules:

```
import os
import time
import count
```

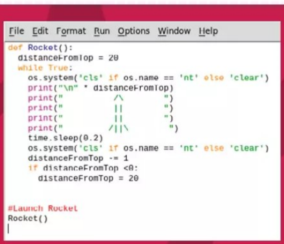
From there, you will need to specify the imported code in the Countdown section:

```
#Start the countdown
for counter in range(start, 0, -1):
    if counter == 10:
        count.ten()
    elif counter == 9:
        count.nine()
    elif counter == 8:
        count.eight()
    elif counter == 7:
        count.seven()
    elif counter == 6:
        count.six()
    elif counter == 5:
        count.five()
    elif counter == 4:
        count.four()
    elif counter == 3:
        count.three()
    elif counter == 2:
        count.two()
    elif counter == 1:
        count.one()
```

This will pull the functions from the imported Count.py and print them to the screen.

ROCKET LAUNCH

Building on the previous countdown example, we can create an animated rocket that'll launch after the Blast Off!! message has been printed. The code would look something like:


File Edit Format Run Options Window Help

```
def Rocket():
  distanceFromTop = 20
  while True:
    os.system('cls' if os.name == 'nt' else 'clear')
    print("\n" * distanceFromTop)
    print("          /\        ")
    print("          ||        ")
    print("          ||        ")
    print("         /||\        ")
    time.sleep(0.2)
    os.system('cls' if os.name == 'nt' else 'clear')
    distanceFromTop -= 1
    if distanceFromTop <0:
      distanceFromTop = 20


#Launch Rocket
Rocket()
```

```
def Rocket():
  distanceFromTop = 20
  while True:
    os.system('cls' if os.name == 'nt' else 'clear')
    print("\n" * distanceFromTop)
    print("          /\        ")
    print("          ||        ")
    print("          ||        ")
    print("         /||\        ")
    time.sleep(0.2)
    os.system('cls' if os.name == 'nt' else 'clear')
    distanceFromTop -= 1
    if distanceFromTop <0:
      distanceFromTop = 20


#Launch Rocket
Rocket()
```

Here, we've created a new function called Rocket, which produces the effect of an ASCII-like rocket taking off and scrolling upwards; using the distanceFromTop variable.

To use this, add it to the end of the previous countdown code and, at the end of the Blast Off!! message, add the following lines:

```
print("\n\n\n\n")
input("Press Enter to launch rocket..."
```

This will allow your message to be displayed and then, when the user has hit the Enter button, the rocket will launch.

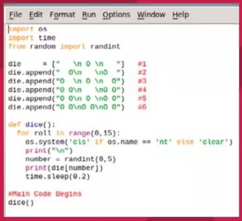
Again, the code in its entirety can be found in the Code Repository at: **https://bdmpublications.com/ code-portal**.

ROLLING DIE

Aside from the rocket animation, together with its countdown, another fun bit of text-based animation is that of a rolling dice.

A rolling dice can be a great animation to include in an adventure game, where the player rolls to see what their score is compared to that of an enemy. The highest roller wins the round and the losers' health drops as a result. It's an age-old combat sequence, used mainly in the Dungeon and Dragons board games and Fighting Fantasy novels, but it works well.

The code you'll need to animate a dice roll is:


File Edit Format Run Options Window Help

```
import os
import time
from random import randint

die      = ["   \n 0 \n    "]   #1
die.append("  0\n   \n0  ")   #2
die.append("0  \n 0 \n  0")   #3
die.append("0 0\n   \n0 0")   #4
die.append("0 0\n 0 \n0 0")   #5
die.append("0 0\n0 0\n0 0")   #6

def dice():
  for roll in range(0,15):
    os.system('cls' if os.name == 'nt' else 'clear')
    print("\n")
    number = randint(0,5)
    print(die[number])
    time.sleep(0.2)

#Main Code Begins
dice()
```

```
import os
import time
from random import randint

die      = ["   \n 0 \n    "]   #1
die.append("  0\n   \n0  ")   #2
die.append("0  \n 0 \n  0")   #3
die.append("0 0\n   \n0 0")   #4
die.append("0 0\n 0 \n0 0")   #5
die.append("0 0\n0 0\n0 0")   #6

def dice():
  for roll in range(0,15):
    os.system('cls' if os.name == 'nt' else
'clear')
    print("\n")
    number = randint(0,5)
    print(die[number])
    time.sleep(0.2)

#Main Code Begins
dice()
```

You may need to tweak the O entries, to line up the dots on the virtual dice. Once it's done, though, you'll be able to add this function to your adventure game code and call it up whenever your character, or the situation, requires some element of luck, combat, or chance roll of the dice.

DISCOVER ANIMATIONS

The great thing about Python code is that it's so accessible. These few examples will help you add some fun, or something different, to your programs, but they're just the tip of the proverbial iceberg. If there's something you want to include in your code, and you're at a sticking point, then consider heading over to Stack Overflow and search for Python 3 content.

Stack Overflow is a great online help and resource portal, where you can ask questions and experts will try and help you. It doesn't always work out, but most of the time, you'll find what you're looking for within this great resource.

Also, if you're after a simple animation then take to Google and spend some time searching for it. While you may not find exactly what it is you're after, you're bound to come across something very like the desired effect; all you need to do is modify it slightly to accomplish your own goals.

You'll find that many professionals will often take to the Internet to find content they're after. True, they're able to code it themselves, but even experts get stuck sometimes, so don't worry about hunting

5 votes 4 answers

Q: How to create ASCII animation in a console application using Python 3.x?

I would like to port this question to **Python** (Windows + Linux + Mac Os) How to create ASCII **animation** in Windows Console application using C#? Thank you! ...

animation python-3.x console-application ascii-art

asked Apr 28 '10 by Hamish Grubijan

0 votes 1 answer

Q: Simple rectangle terminal animation in Python 3.x

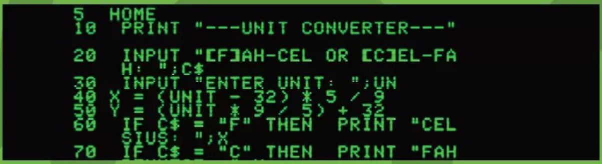
I hava an assignment where I have to write a function that will print a rectangle (user can change it's height a width) frame by frame with a square of "x""es inside of it (I tried my best to descri ...

python arrays python-3.x list

asked Oct 28 '17 by Dartosz K

Retro Coding

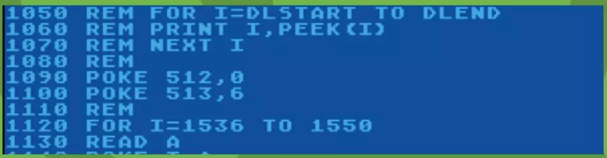
There's a school of thought, that to master the foundations of good coding skills you need to have some experience of how code was written in the past. In the past is a bit of a loose term, but mostly, it means coding from the 80s.



```
5  HOME
10  PRINT "---UNIT CONVERTER---"
20  INPUT "[F]AH-CEL OR [C]EL-FA
    H: ";C$
30  INPUT "ENTER UNIT: ";UN
40 X = (UNIT - 32) * 5 / 9
50 Y = (UNIT * 9 / 5) + 32
60  IF C$ = "F" THEN  PRINT "CEL
    SIUS: ";X
70  IF C$ = "C" THEN  PRINT "FAH
```

THE GOLDEN ERA OF CODE

While it may seem a little counterproductive to learn how to code in a language that's virtually obsolete, there are some surprising benefits to getting your hands dirty with a bit of retro coding. Firstly, learning old code will help you build the structure of code as, regardless of whether it is a language that was developed yesterday, or forty years ago, code still demands strict discipline to work correctly. Secondly, everyday coding elements, such as loops, sub routines and so on, are a great visual aid to learn in older code, especially BASIC. Lastly, it's simply good fun.



```
1050 REM FOR I=DLSTART TO DLEND
1060 REM PRINT I,PEEK(I)
1070 REM NEXT I
1080 REM
1090 POKE 512,0
1100 POKE 513,6
1110 REM
1120 FOR I=1536 TO 1550
1130 READ A
```

GOING BASIC

The easiest retro language to play around with is, without doubt, BASIC. Developed back in the mid-sixties, BASIC (Beginner's All-Purpose Symbolic Instruction Code) is a high-level programming language whose design was geared toward ease of use. In a time when computers were beginning to become more accessible, designers John Kemeny and Thomas Kurtz needed a language that students could get to grips with, quickly and easily. Think of BASIC as a distant relation to Python.

THE BEEB

The problem with BASIC is that there were so many different versions available, across multiple 8-bit platforms, with each having its own unique elements on top of the core BASIC functions. The BASIC that was packaged with the Commodore 64 was different to that on the ZX Spectrum, or the Atari home computers, due to the differing hardware of each system. However, it's widely recognised that one of the 'best', and possibly most utilised, form of BASIC from the 80s was that of BBC BASIC.

BBC BASIC was used on the Acorn BBC Micro range of computers, utilising the MOS 6502-based processor technologies. It was one of the quickest examples of BASIC and, thanks to an inline assembler, it was also capable of allowing the developers of the time to write code for different processor types, such as the Zilog Z80 – a CPU present in the ZX Spectrum, as well as many arcade machines.

The BBC Micro was designed and built by Acorn Computers – a company that is historically responsible for the creation of the ARM CPU - the processor that's used in virtually every Android phone and tablet, smart TV, set top box and so on, as well as the Raspberry Pi – so essentially, the BBC Micro is the grandfather of the Raspberry Pi.

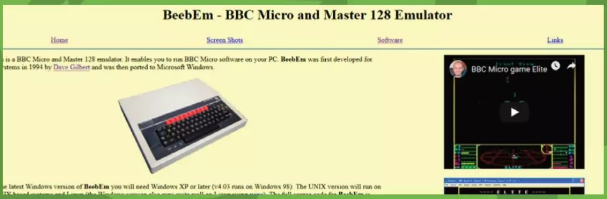
The BBC Micro was born in a time when the UK government was looking for a countrywide computer platform to be used throughout education. Different companies bid, but it was the BBC's Computer Literacy Project (the BBC Micro) that was chosen, due to its ruggedness, upgradability, and potential for education. As a result, the BBC Micro, or the Beeb as it's affectionately known, became the dominant educational computer throughout the 80s.

BEEBEM

Naturally, you could scour eBay and look for a working BBC Micro to play around on, and it'll be a lot of fun. However, for the sake of just getting hands-on with some retro code, we'll use one of the best BBC Micro emulators available: BeebEm.

BeebEm was originally developed for UNIX in 1994 by Dave Gilbert and later ported to Windows. It is now developed by Mike Wyatt and Jon Welch, who maintain the Mac port of the emulator, and is therefore available for Windows 10, Linux and macOS, as well as other platforms.

If you're using Windows 10, simply navigate to **http://www.mkw.me.uk/beebem/index.html**, and download the BeebEM414.exe that's displayed in the main screen.



BeebEm - BBC Micro and Master 128 Emulator

Home Screen Shots Software Links

is a BBC Micro and Master 128 emulator. It enables you to run BBC Micro software on your PC. **BeebEm** was first developed for stems in 1994 by Dave Gilbert and was then ported to Microsoft Windows.

e latest Windows version of **BeebEm** you will need Windows XP or later (v4.03 runs on Windows 98). The UNIX version will run on

Once downloaded, launch the executable and follow the on-screen instructions to install it. MacOS users can get everything they need from: **http://www.g7jjf.com/**. However, Raspberry Pi and Linux users will have to do a little nifty keyboard work before they can enjoy the benefits of the Beeb on their screens. Here's how to get it working under Linux:

First, drop to a Terminal and enter:

```
sudo apt-get update && upgrade
wget http://beebem-unix.bbcmicro.com/download/
beebem-0.0.13.tar.gz
```

Then, extract the compressed files with:

`tar zxvf beebem-0.0.13.tar.gz`, then enter the newly created Beebem folder with: `cd beebem-0.0.13/.`

Now enter the following lines, hitting Enter and answering Y to accept any changes:

```
sudo apt-get install libgtk2.0-dev libsdl1.2-dev
./configure
make
sudo make install
```

This may take some time, but when it's all done, simply enter: **beebem**, to start the BBC Micro emulator.

BBC BASIC

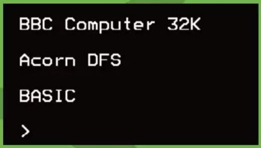
Once installed and powered up, BeebEm will display the default BBC system start-up, along with a couple of beeps. Those of you old enough to have been in a UK school in the 80s will certainly recall this setup.

In BASIC, we use line numbers to determine which lines of code run in sequence. For example, to print something to screen we'd enter:

```
10 print "hello"
```

Once you've typed the above in, press Enter and then type:

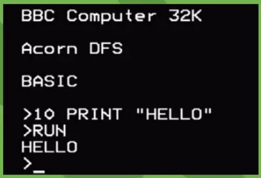
```
run
```



```
BBC Computer 32K

Acorn DFS

BASIC

>
```

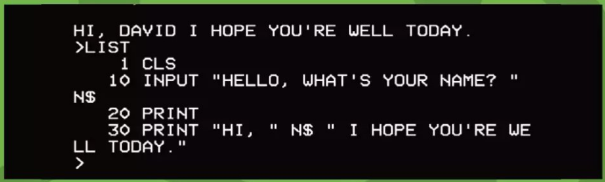
We can of course expand the code to include variables, multi-line print statements, and so on:

```
1 CLS
10 Input "Hello, what's your name? " n$
20 print
30 print "Hi, " n$ " I hope
you're well today."
```



```
BBC Computer 32K

Acorn DFS

BASIC

>10 PRINT "HELLO"
>RUN
HELLO
>_
```

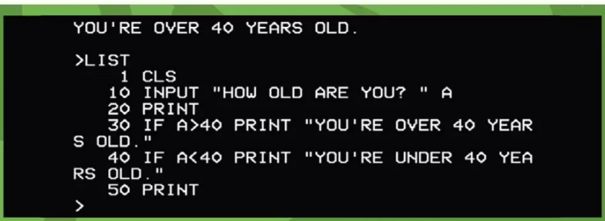
Type RUN to execute the code, you can also type LIST to view the code you've entered.



```
HI, DAVID I HOPE YOU'RE WELL TODAY.
>LIST
    1 CLS
   10 INPUT "HELLO, WHAT'S YOUR NAME? "
N$
   20 PRINT
   30 PRINT "HI, " N$ " I HOPE YOU'RE WE
LL TODAY."
>
```

As you can see, variables are handled in much the same way as Python, a print statement on its own displays a blank line, and CLS clears the screen; although the Pi uses Clear instead of CLS. We're also able to do some maths work, and play around with variables too:

```
1 CLS
10 input "how old are you? " a
20 print
30 if a > 40 print "You're over 40 years old."
40 if a < 40 print "You're under 40 years old."
```

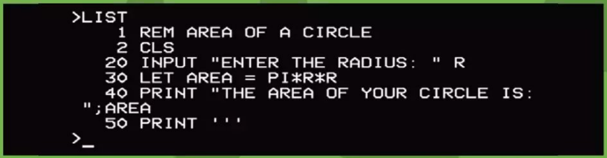


```
YOU'RE OVER 40 YEARS OLD.

>LIST
    1 CLS
   10 INPUT "HOW OLD ARE YOU? " A
   20 PRINT
   30 IF A>40 PRINT "YOU'RE OVER 40 YEAR
S OLD."
   40 IF A<40 PRINT "YOU'RE UNDER 40 YEA
RS OLD."
   50 PRINT
>
```

```
50 print
```

BBC BASIC also has some interesting built-in features, such as the value of PI:

```
1 REM Area of a circle
2 CLS
20 Input "Enter the radius: " r
30 let area = PI*r*r
40 print "The area of your circle is: "; area
50 print ''
```



```
>LIST
    1 REM AREA OF A CIRCLE
    2 CLS
   20 INPUT "ENTER THE RADIUS: " R
   30 LET AREA = PI*R*R
   40 PRINT "THE AREA OF YOUR CIRCLE IS:
 ";AREA
   50 PRINT ''
>_
```

As you'll notice, variables with a dollar ($) represent strings, nothing after the variable, or a hash (#) represent floating point decimals; a whole integer has a % character, and a byte has an ampersand (&). The single quotes after the Print on line 50 indicate a blank line, one for each tick, while REM on line 1 is a comment, and thus ignored by the BASIC compiler.

Needless to say, there's a lot you can learn, as well as having fun, with BBC BASIC. It's a rainy day project and something that's interesting to show the kids – this is how we rolled back in the 80s, kids!

There are a number of sites you can visit to learn BBC BASIC, such as **http://archive.retro-kit.co.uk/bbc.nvg.org/docs.php3.html**. See what you can come up with using BBC BASIC, or other BASIC types for different systems, and let us know what you've created.

OTHER SYSTEMS

128 BASIC
Calculator
48 BASIC
Tape Tester
© 1986 Sinclair Research
Amstrad 64K Microcomputer (v1)
BASIC 1.0
Ready
**** COMMODORE 64 BASIC V2 ****
READY.
GEM, Graphics Environment Manager

Naturally, you don't have to look to the BBC Micro to play around with some retro code. If you grew up with a Commodore 64, then you can always try VICE, the C64 emulator. Likewise, the ZX Spectrum has a slew of great emulators available for every modern system to play around on. In fact, you can probably find an emulator for virtually every 8-bit or 16-bit machine that was produced over the years. Each has their own unique perspective and coding nuances, so find a few and see what you can create.

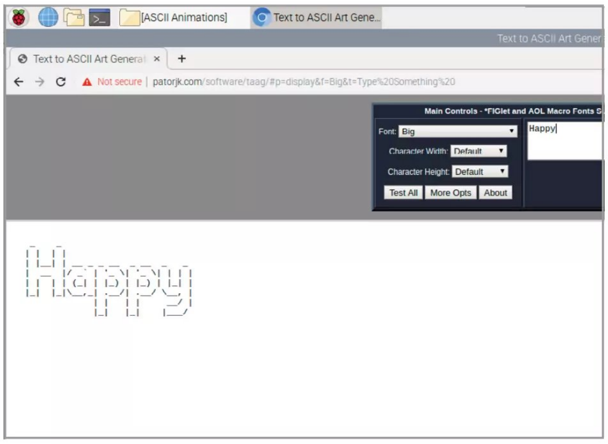
Using Text Files for Animation

Animation in Python can be handled with the likes of the Tkinter and Pygame modules, however, there's more than one way to achieve a decent end result. Using some clever, text file reading code, we can create command-line animations.

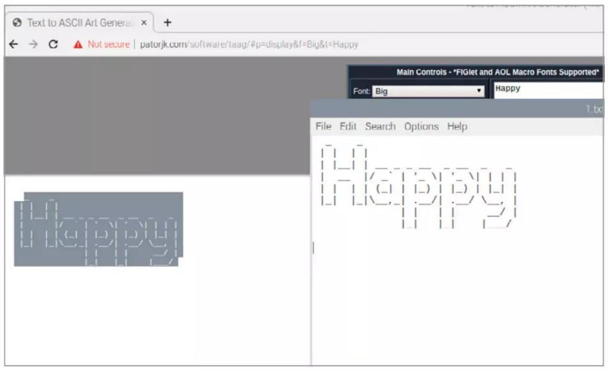
ASCII ANIMATION

Let's assume you wanted to create an animated ASCII Happy Birthday Python script, with the words Happy and Birthday alternating in appearance. Here's how it's done.

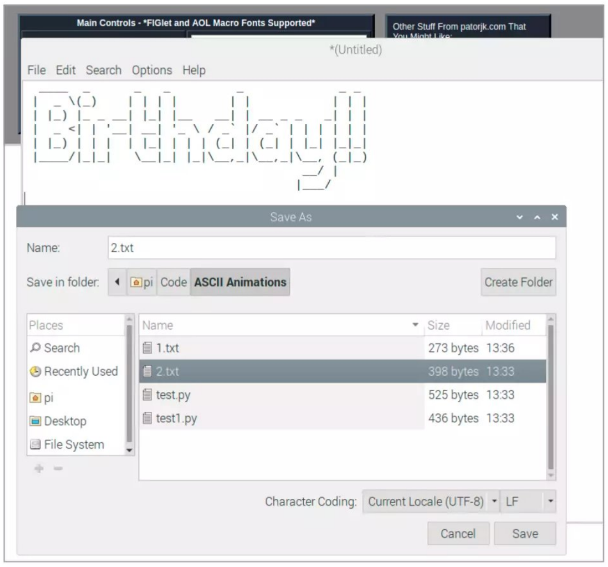
STEP 1 First we need to create some ASCII-like text, head over to **http://patorjk.com/software/taag**. This is an online Text to ASCII generator, created by Patrick Gillespie. Start by entering **Happy** into the text box, the result will be displayed in the main window. You can change the font with the drop-down menu to the side of the text box; we've opted for **Big**.


[ASCII Animations]
Text to ASCII Art Gene...
Not secure | patorjk.com/software/taag/#p=display&f=Big&t=Type%20Something%20
Main Controls - *FIGlet and AOL Macro Fonts S
Font: Big
Character Width: Default
Character Height: Default
Test All
More Opts
About
Happy

STEP 2 Now, on your computer create a folder in your Python code directory (call it **Test**, for now) and open either Leafpad for the Raspberry Pi or, if you're using Windows 10, then Notepad. Click on the **Select & Copy** button, at the bottom of the ASCII Generator webpage, and paste the contents into the text editor.


Text to ASCII Art Genera
Not secure | patorjk.com/software/taag/#p=display&f=Big&t=Happy
Main Controls - *FIGlet and AOL Macro Fonts Supported*
Font: Big
Happy
1.txt
File Edit Search Options Help

STEP 3 Save the text file as **1.txt** (you can call it anything, but now for ease of use 1.txt will suffice) and save the file in the newly created Test folder. When it's saved, do exactly the same for the word Birthday. You can select a new font from the ASCII Generator, or add extra characters, and when you're ready save the file as **2.txt**.


Main Controls - *FIGlet and AOL Macro Fonts Supported*
Other Stuff From patorjk.com That You Might Like:
*(Untitled)
File Edit Search Options Help
Save As
Create Folder
Places
Search
Recently Used
pi
Desktop
File System
Cancel
Save

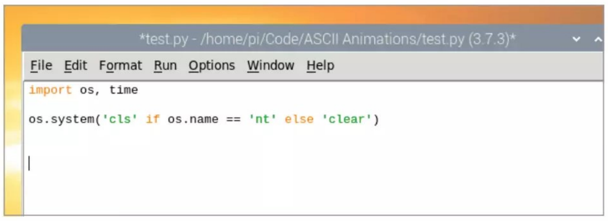
Name: 2.txt

Save in folder: pi | Code | **ASCII Animations**

Name	Size	Modified
1.txt	273 bytes	13:36
2.txt	398 bytes	13:33
test.py	525 bytes	13:33
test1.py	436 bytes	13:33

Character Coding: Current Locale (UTF-8) | LF

STEP 4 Open up Python and create a **New File**. We're going to need to import the OS and Time modules for this example, followed by a line to clear the screen of any content. If you're using Windows, then you'll use the CLS command, whereas it's Clear for the Raspberry Pi's Linux OS. We can create a simple if/else statement to handle the command.

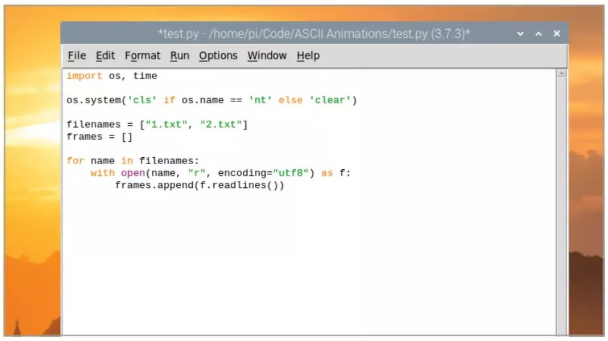

test.py - /home/pi/Code/ASCII Animations/test.py (3.7.3)
File Edit Format Run Options Window Help

```
import os, time

os.system('cls' if os.name == 'nt' else 'clear')
```

STEP 5

Next we need to create a list with the names of the text files we want to open, and then we need to open them for display in the Terminal.

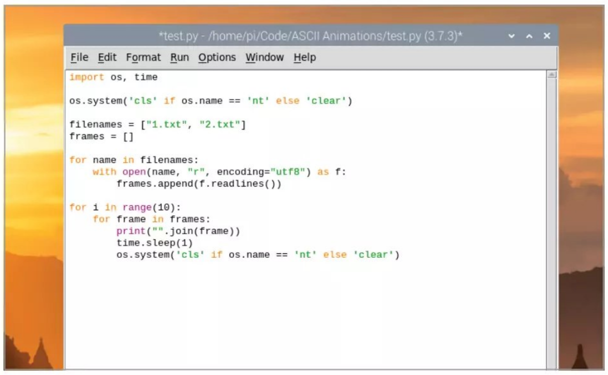
```
filenames = ["1.txt", "2.txt"]
frames = []

for name in filenames:
    with open(name, "r", encoding="utf8") as f:
        frames.append(f.readlines())
```


test.py - /home/pi/Code/ASCII Animations/test.py (3.7.3)
File Edit Format Run Options Window Help

```
import os, time

os.system('cls' if os.name == 'nt' else 'clear')

filenames = ["1.txt", "2.txt"]
frames = []

for name in filenames:
    with open(name, "r", encoding="utf8") as f:
        frames.append(f.readlines())
```

STEP 6

We've used the UTF8 standard when opening the text files, as ASCII art as text, within a text file, often requires you to save the file as UTF compliant – due to the characters used. Now we can add a loop to display the files as 1.txt, then 2.txt, creating the illusion of animation while clearing the screen after each file is displayed.


test.py - /home/pi/Code/ASCII Animations/test.py (3.7.3)
File Edit Format Run Options Window Help

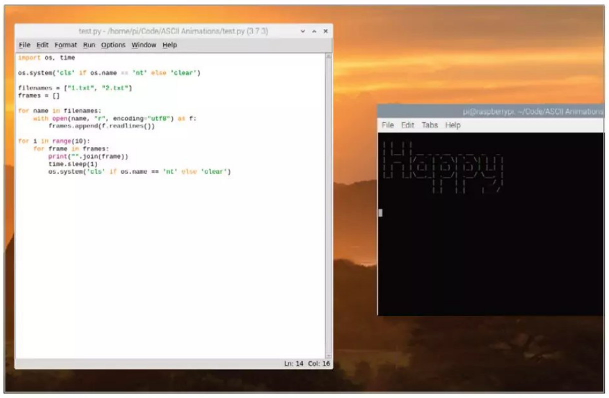
```
import os, time

os.system('cls' if os.name == 'nt' else 'clear')

filenames = ["1.txt", "2.txt"]
frames = []

for name in filenames:
    with open(name, "r", encoding="utf8") as f:
        frames.append(f.readlines())

for i in range(10):
    for frame in frames:
        print("".join(frame))
        time.sleep(1)
        os.system('cls' if os.name == 'nt' else 'clear')
```

STEP 7

Save the Python code in the same folder as the text files and drop into a Terminal or Command Prompt. Navigate to the folder in question, and enter the command:

```
python3 NAME.py
```

Where NAME is whatever you called your saved Python code.


test.py - /home/pi/Code/ASCII Animations/test.py (3.7.3)
File Edit Format Run Options Window Help
pi@raspberrypi: ~/Code/ASCII Animations
File Edit Tabs Help
Happy
Ln: 14 Col: 16

```
import os, time

os.system('cls' if os.name == 'nt' else 'clear')

filenames = ["1.txt", "2.txt"]
frames = []

for name in filenames:
    with open(name, "r", encoding="utf8") as f:
        frames.append(f.readlines())

for i in range(10):
    for frame in frames:
        print("".join(frame))
        time.sleep(1)
        os.system('cls' if os.name == 'nt' else 'clear')
```

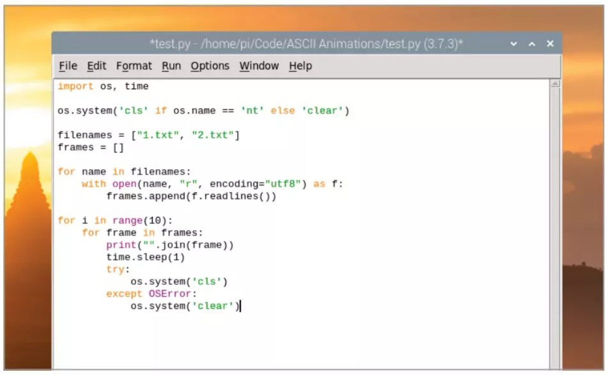
STEP 8

Here's the code in full:

```
import os, time

os.system('cls' if os.name == 'nt' else 'clear')


filenames = ["1.txt", "2.txt"]
frames = []

for name in filenames:
    with open(name, "r", encoding="utf8") as f:
        frames.append(f.readlines())

for i in range(10):
    for frame in frames:
        print("".join(frame))
        time.sleep(1)
        os.system('cls' if os.name == 'nt' else 
'clear')
```

STEP 9

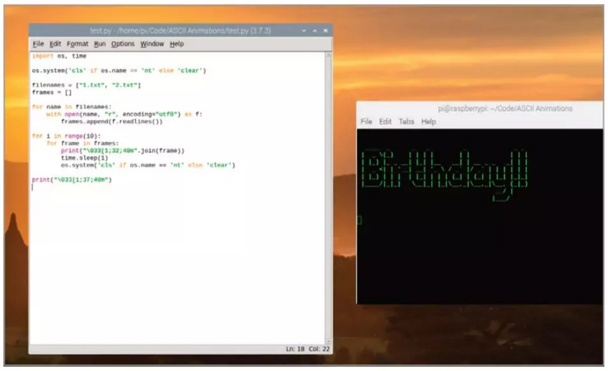
Note, from the loop, within the code, we've used the same CLS and Clear if/else statement as before. Again, if you're running on Windows then the OS module will use the CLS command, 'ELSE' if you're using Linux or a Mac, the Clear command will work correctly. If you want, you could use a Try/Except statement instead.


test.py - /home/pi/Code/ASCII Animations/test.py (3.7.3)
File Edit Format Run Options Window Help

```
import os, time

os.system('cls' if os.name == 'nt' else 'clear')

filenames = ["1.txt", "2.txt"]
frames = []

for name in filenames:
    with open(name, "r", encoding="utf8") as f:
        frames.append(f.readlines())

for i in range(10):
    for frame in frames:
        print("".join(frame))
        time.sleep(1)
        try:
            os.system('cls')
        except OSError:
            os.system('clear')
```

STEP 10

You can spice things up a little by adding system/Terminal colours. You'll need to Google the system codes for the colours you want. The code in our example turns the Raspberry Pi Terminal to green text on a black background, then changes it back to white on black at the end of the code. Either way, it's a fun addition to your Python code.


test.py - /home/pi/Code/ASCII Animations/test.py (3.7.3)
File Edit Format Run Options Window Help
pi@raspberrypi: ~/Code/ASCII Animations
File Edit Tabs Help
Birthday!!
Ln: 18 Col: 22

```
import os, time

os.system('cls' if os.name == 'nt' else 'clear')

filenames = ["1.txt", "2.txt"]
frames = []

for name in filenames:
    with open(name, "r", encoding="utf8") as f:
        frames.append(f.readlines())

for i in range(10):
    for frame in frames:
        print("\033[1;32;40m".join(frame))
        time.sleep(1)
        os.system('cls' if os.name == 'nt' else 'clear')

print("\033[1;37;40m")
```

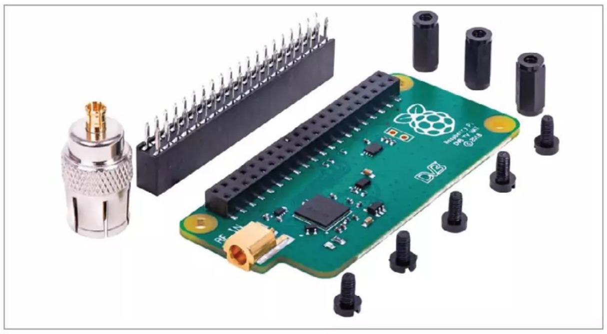
Stream Digital TV with a HAT - Part 1

A HAT, in case you're wondering, is a Hardware Attached on Top add-on board that connects to a Raspberry Pi's 40-pin GPIO. They can extend the capabilities of a Pi by adding motors, LCDs, sensors, controllers and more. In addition, they can be programmed via the Pi and Python.

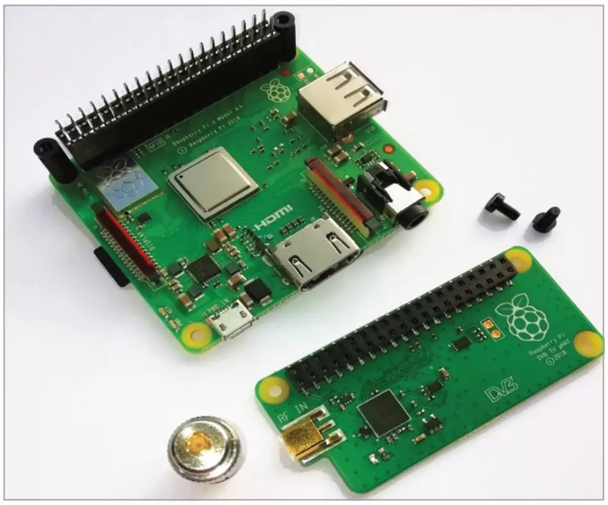
TV HAT

In this tutorial, we're using the Raspberry Pi TV HAT as sold by The Pi Hut (https://thepihut.com/products/raspberry-pi-tv-hat?variant=13539182673982). It's a Sony CXD2880 TV Tuner supporting DVB-T and 2nd-gen DVB-T2 standards.

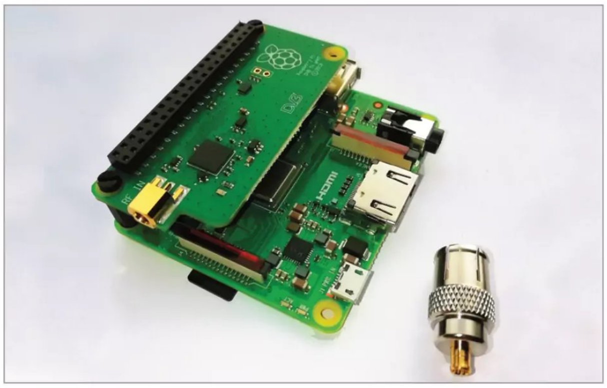
STEP 1 The Raspberry Pi TV HAT is compatible with the Pi Zero, Pi 3 A+ and B+ models. The kit comes with everything you need to connect the HAT to your RPi, including spacers, screws and the 40-pin header. Begin by opening the box and spreading the contents out on a clear area.



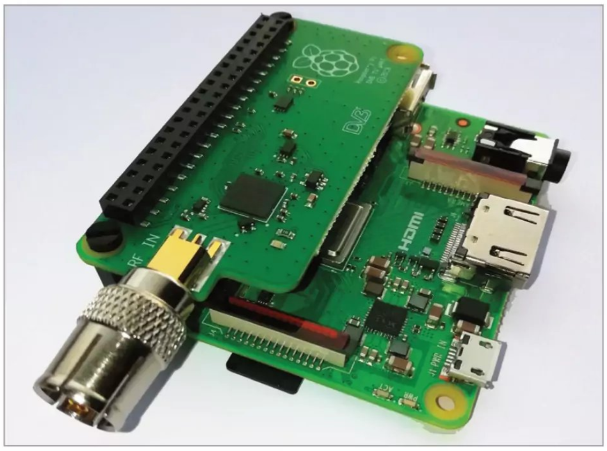
STEP 2 Start by connecting the 40-pin header to the 40-pin GPIO on top of your Raspberry Pi. Now take the spacers and screws and fit them to the corresponding holes in the corners of the main Raspberry Pi and the TV HAT; use three spacers and screws for the Pi Zero and two for the Pi 3 Models A+/B+.



STEP 3 With the spacers attached to the Raspberry Pi, line up the TV HAT with the 40-pin header, ensuring that the side of the HAT with the Pi logo is facing up and the HAT's gold coloured coaxial attachment is on the side of the SD card. When in place, screw the HAT to its spacers.



STEP 4 Now attach the coaxial adapter to the gold-coloured pin at the side of the TV HAT. It may need a firm push to lock into place, it will 'click' when fully inserted and in position. Now you will need to connect the TV HAT to your TV aerial and provide power to the Raspberry Pi.



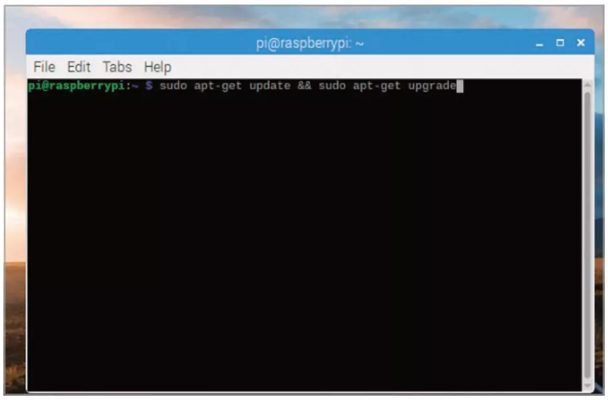
STEP 5

You can use the HDMI port to connect your Pi and its TV HAT to the TV/monitor, or you can just power the Pi (the TV HAT gets its power from the Pi) and connect remotely. The most important element is to ensure that the TV HAT is connected to an aerial that you know can receive a TV signal.

STEP 6

Power up your Raspberry Pi and when you get to the Raspbian desktop, open up the Terminal app. When at the Terminal, enter the following to ensure the Pi is up to date and running the latest versions of its software and system:

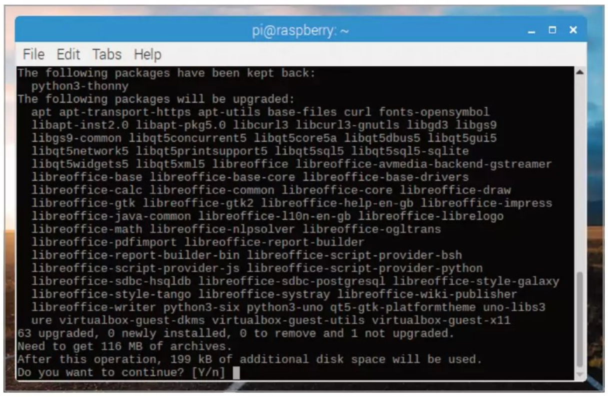
```
sudo apt-get update && sudo apt-get upgrade
```



```
pi@raspberrypi:~ $ sudo apt-get update && sudo apt-get upgrade
```

STEP 7

Press Enter and allow the Pi to run through the update process. If you have any significant updates, you may need to answer 'Y' to any questions the Pi asks regarding these. Answering yes will replace the older software with the newer versions and is necessary for a smoothly running Pi.



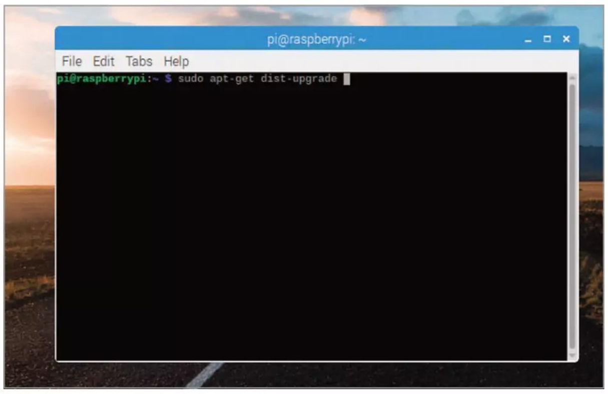
```
The following packages have been kept back:
  python3-thonny
The following packages will be upgraded:
  apt apt-transport-https apt-utils base-files curl fonts-opensymbol
  libapt-inst2.0 libapt-pkg5.0 libcurl3 libcurl3-gnutls libgd3 libgs9
  libgs9-common libqt5concurrent5 libqt5core5a libqt5dbus5 libqt5gui5
  libqt5network5 libqt5printsupport5 libqt5sql5 libqt5sql5-sqlite
  libqt5widgets5 libqt5xml5 libreoffice libreoffice-avmedia-backend-gstreamer
  libreoffice-base libreoffice-base-core libreoffice-base-drivers
  libreoffice-calc libreoffice-common libreoffice-core libreoffice-draw
  libreoffice-gtk libreoffice-gtk2 libreoffice-help-en-gb libreoffice-impress
  libreoffice-java-common libreoffice-l10n-en-gb libreoffice-librelogo
  libreoffice-math libreoffice-nlpsolver libreoffice-ogltrans
  libreoffice-pdfimport libreoffice-report-builder
  libreoffice-report-builder-bin libreoffice-script-provider-bsh
  libreoffice-script-provider-js libreoffice-script-provider-python
  libreoffice-sdbc-hsqldb libreoffice-sdbc-postgresql libreoffice-style-galaxy
  libreoffice-style-tango libreoffice-systray libreoffice-wiki-publisher
  libreoffice-writer python3-six python3-uno qt5-gtk-platformtheme uno-libs3
  ure virtualbox-guest-dkms virtualbox-guest-utils virtualbox-guest-x11
63 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
Need to get 116 MB of archives.
After this operation, 199 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

STEP 8

If you haven't used your Raspberry Pi since, at least, November 2018, then you may need to upgrade the core OS and synchronise the version of Raspbian with what's currently available from the Pi Foundation's downloads page. It's not totally necessary, but if you choose to, enter:

```
sudo apt-get dist-upgrade
```

Press Enter and follow the on-screen instructions.



```
pi@raspberrypi:~ $ sudo apt-get dist-upgrade
```

STEP 9

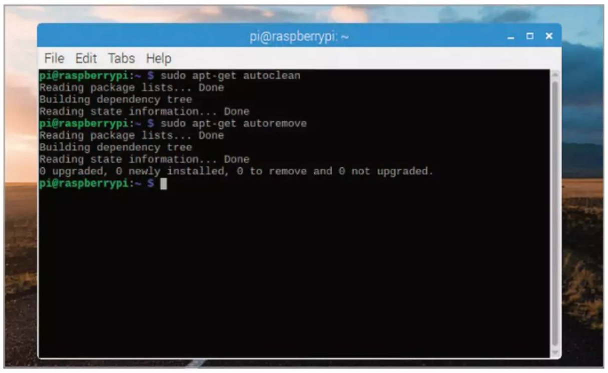
When the updates are complete, there's a good chance you've got some leftover, older setup files and packages in the system. To save space, use the following commands:

```
sudo apt-get autoclean
```

and:

```
sudo apt-get autoremove
```

Answer 'Y' to clear the system of the unnecessary packages and files.



```
pi@raspberrypi:~ $ sudo apt-get autoclean
Reading package lists... Done
Building dependency tree
Reading state information... Done
pi@raspberrypi:~ $ sudo apt-get autoremove
Reading package lists... Done
Building dependency tree
Reading state information... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
pi@raspberrypi:~ $
```

STEP 10

At this point, it's worth noting that in the UK it's necessary to have a valid TV License in order to watch or record programmes as they are being shown on TV or live via an online service. It is an offence under Section 363 of the Communications Act 2003 to use a TV receiving device without a valid TV License.

Stream Digital TV with a HAT - Part 2

In part 1 of this tutorial, we set up the Raspberry Pi and the TV HAT. The core Pi software and system should now be up to date and the TV Hat connected to an aerial. You should also ensure you have a valid UK TV Licence.

TURN ON, TUNE IN

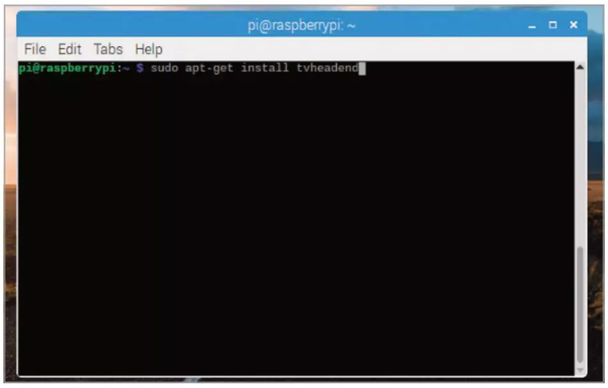
With the hardware ready, it's now time to begin the TV tuning software installation and setup. If you haven't already, reboot the Raspberry Pi and enter the Terminal for this next part.

STEP 1

To use the TV tuner, we need to install the TVheadend software. Open a Terminal session and enter the following:

`sudo apt-get install tvheadend`

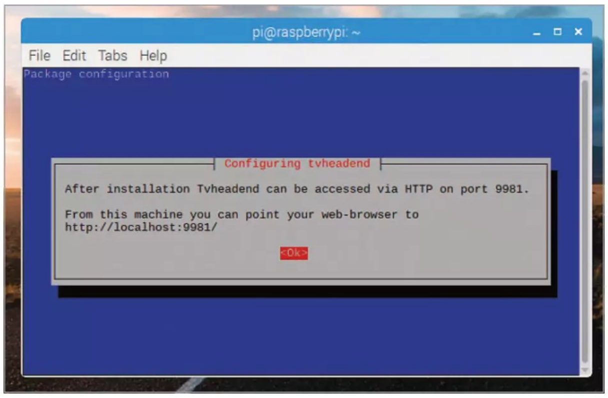
Press Enter and 'Y' if necessary to confirm the installation.



```
pi@raspberrypi:~ $ sudo apt-get install tvheadend
```

STEP 2

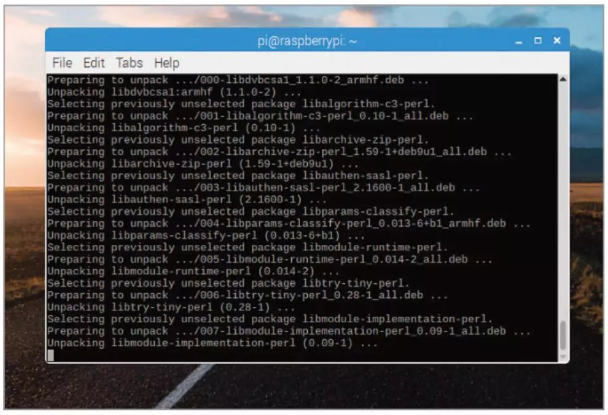
As the installation continues, you will be presented with a configuration screen. Enter a username and password to enable access to the TVheadend server. Once entered, make a note of the web address access for the server. When accessing on the TV Pi, it'll be **http://localhost:9981/**. If accessing from another computer on the network, use the Pi's IP address. For example: **http://192.168.1.223:9981/**.



```
Package configuration

Configuring tvheadend

After installation Tvheadend can be accessed via HTTP on port 9981.

From this machine you can point your web-browser to
http://localhost:9981/

<Ok>
```

STEP 3

The remainder of the setup will now continue. It'll take around three minutes to complete, depending on which model Raspberry Pi you're using. When the setup has finished, you can exit the Terminal session.



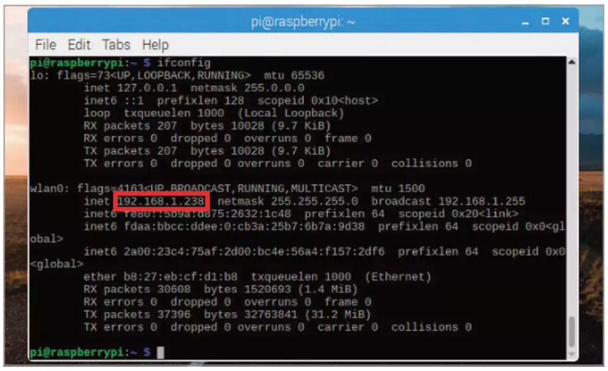
```
Preparing to unpack .../000-libdvbcsa1_1.1.0-2_armhf.deb ...
Unpacking libdvbcsa1:armhf (1.1.0-2) ...
Selecting previously unselected package libalgorithm-c3-perl.
Preparing to unpack .../001-libalgorithm-c3-perl_0.10-1_all.deb ...
Unpacking libalgorithm-c3-perl (0.10-1) ...
Selecting previously unselected package libarchive-zip-perl.
Preparing to unpack .../002-libarchive-zip-perl_1.59-1+deb9u1_all.deb ...
Unpacking libarchive-zip-perl (1.59-1+deb9u1) ...
Selecting previously unselected package libauthen-sasl-perl.
Preparing to unpack .../003-libauthen-sasl-perl_2.1600-1_all.deb ...
Unpacking libauthen-sasl-perl (2.1600-1) ...
Selecting previously unselected package libparams-classify-perl.
Preparing to unpack .../004-libparams-classify-perl_0.013-6+b1_armhf.deb ...
Unpacking libparams-classify-perl (0.013-6+b1) ...
Selecting previously unselected package libmodule-runtime-perl.
Preparing to unpack .../005-libmodule-runtime-perl_0.014-2_all.deb ...
Unpacking libmodule-runtime-perl (0.014-2) ...
Selecting previously unselected package libtry-tiny-perl.
Preparing to unpack .../006-libtry-tiny-perl_0.28-1_all.deb ...
Unpacking libtry-tiny-perl (0.28-1) ...
Selecting previously unselected package libmodule-implementation-perl.
Preparing to unpack .../007-libmodule-implementation-perl_0.09-1_all.deb ...
Unpacking libmodule-implementation-perl (0.09-1) ...
```

STEP 4

You can now either open the web browser on the Pi, if you're connected to the TV through the Pi, or, if you're connecting remotely, open any web browser on your computer with the address from Step 2. If you don't know the Pi's IP address enter:

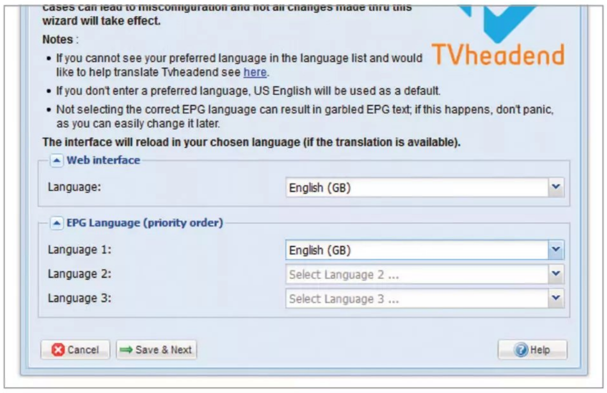
`ifconfig`

in the Terminal on the Pi. The **inet** entry is the Pi's IP address. Here, the example is: **192.168.1.238**.



```
pi@raspberrypi:~ $ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 207  bytes 10028 (9.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 207  bytes 10028 (9.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.238  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::5b9a:d875:2632:1c48  prefixlen 64  scopeid 0x20<link>
        inet6 fdaa:bbcc:ddee:0:cb3a:25b7:6b7a:9d38  prefixlen 64  scopeid 0x0<gl
obal>
        inet6 2a00:23c4:75af:2d00:bc4e:56a4:f157:2df6  prefixlen 64  scopeid 0x0
<global>
        ether b8:27:eb:cf:d1:b8  txqueuelen 1000  (Ethernet)
        RX packets 30608  bytes 1520693 (1.4 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 37396  bytes 32763841 (31.2 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

pi@raspberrypi:~ $
```

STEP 5 For this example, we will assume you're connecting remotely (from another computer on the home network). Enter the Pi's IP address with the port 9981, e.g.: **http://192.168.1.238:9981/**. To start the Configuration Wizard: enter the username and password you set up from Step 2 and log in to the TVheadend Server, then set the Language and Language 1 options to your preference.



cases can lead to misconfiguration and not all changes made thru this wizard will take effect.

Notes :

- If you cannot see your preferred language in the language list and would like to help translate Tvheadend see here.
- If you don't enter a preferred language, US English will be used as a default.
- Not selecting the correct EPG language can result in garbled EPG text; if this happens, don't panic, as you can easily change it later.

The interface will reload in your chosen language (if the translation is available).

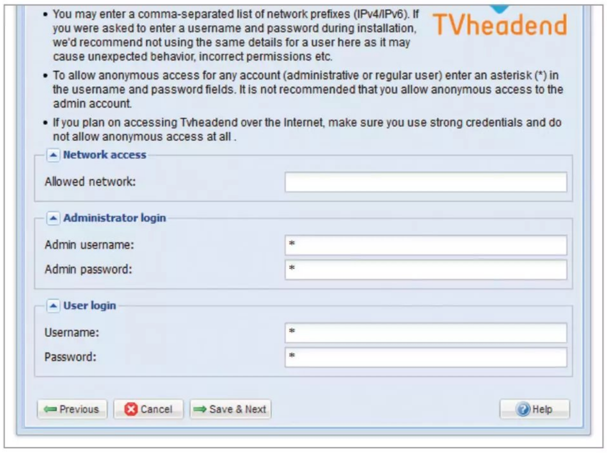
Web interface

Language:	English (GB)

EPG Language (priority order)

Language 1:	English (GB)
Language 2:	Select Language 2 ...
Language 3:	Select Language 3 ...

STEP 6 Click the **Save & Next** button to continue. You will need to enable network access to the server. Leave the Allowed Network field blank, but add an asterisk (*) in each of the other fields. Click **Save & Next** for the next step in the setup process.



- You may enter a comma-separated list of network prefixes (IPv4/IPv6). If you were asked to enter a username and password during installation, we'd recommend not using the same details for a user here as it may cause unexpected behavior, incorrect permissions etc.
- To allow anonymous access for any account (administrative or regular user) enter an asterisk (*) in the username and password fields. It is not recommended that you allow anonymous access to the admin account.
- If you plan on accessing Tvheadend over the Internet, make sure you use strong credentials and do not allow anonymous access at all .

Network access

Allowed network:	

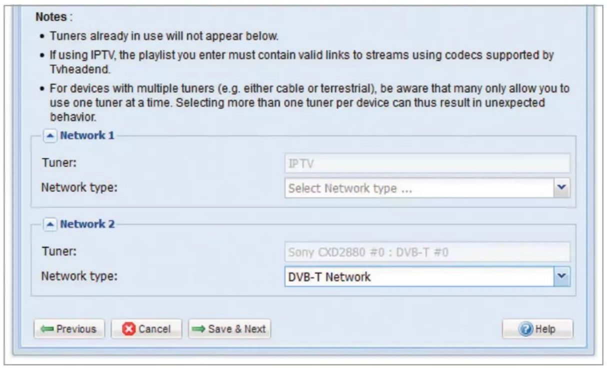
Administrator login

Admin username:	*
Admin password:	*

User login

Username:	*
Password:	*

STEP 7 For the Network Settings page, leave the first three fields blank, but use the pull-down menu to select DVB-T Network. The TVheadend server will already have pre-selected the Sony CXD2880 tuner (the TV HAT) from its available choices. Click the **Save & Next** button when ready.



Notes :

- Tuners already in use will not appear below.
- If using IPTV, the playlist you enter must contain valid links to streams using codecs supported by Tvheadend.
- For devices with multiple tuners (e.g. either cable or terrestrial), be aware that many only allow you to use one tuner at a time. Selecting more than one tuner per device can thus result in unexpected behavior.

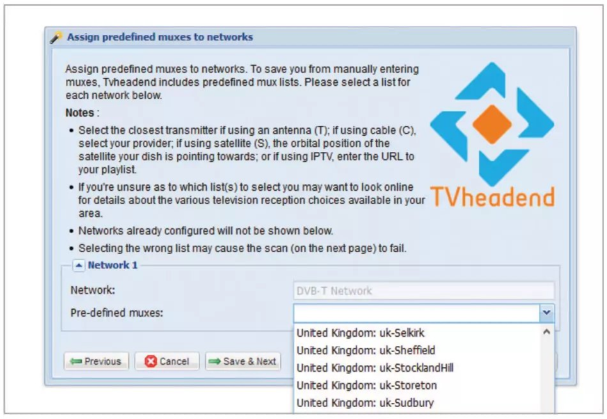
Network 1

Tuner:	IPTV
Network type:	Select Network type ...

Network 2

Tuner:	Sony CXD2880 #0 : DVB-T #0
Network type:	DVB-T Network

STEP 8 For this next step you need to select the transmitter closest to your location. You can find the closest transmitter by entering your details at: **http://www.digitaluk.co.uk/coveragechecker/**. Simply choose the transmitter from the pull-down list. When ready, click on the **Save & Next** button. The TV HAT will now scan for all available TV signals from the chosen transmitter.



Assign predefined muxes to networks

Assign predefined muxes to networks. To save you from manually entering muxes, Tvheadend includes predefined mux lists. Please select a list for each network below.

Notes :

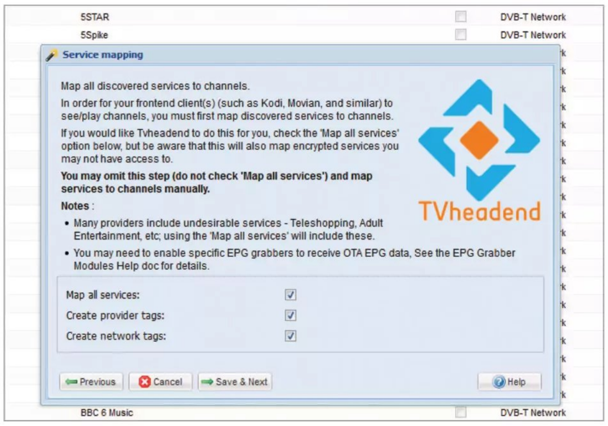
- Select the closest transmitter if using an antenna (T); if using cable (C), select your provider; if using satellite (S), the orbital position of the satellite your dish is pointing towards; or if using IPTV, enter the URL to your playlist.
- If you're unsure as to which list(s) to select you may want to look online for details about the various television reception choices available in your area.
- Networks already configured will not be shown below.
- Selecting the wrong list may cause the scan (on the next page) to fail.

Network 1

Network:	DVB-T Network
Pre-defined muxes:	

United Kingdom: uk-Selkirk
United Kingdom: uk-Sheffield
United Kingdom: uk-StocklandHill
United Kingdom: uk-Storeton
United Kingdom: uk-Sudbury

STEP 9 On the next screen, tick all three of the available boxes. Click the **Save & Next** box followed by the **Finish** button on the next screen; It's recommended that you now reboot the Pi and when it's fully rebooted, navigate back to the TVheadend server webpage.



5STAR DVB-T Network
5Spike DVB-T Network

Service mapping

Map all discovered services to channels.

In order for your frontend client(s) (such as Kodi, Movian, and similar) to see/play channels, you must first map discovered services to channels.

If you would like Tvheadend to do this for you, check the 'Map all services' option below, but be aware that this will also map encrypted services you may not have access to.

You may omit this step (do not check 'Map all services') and map services to channels manually.

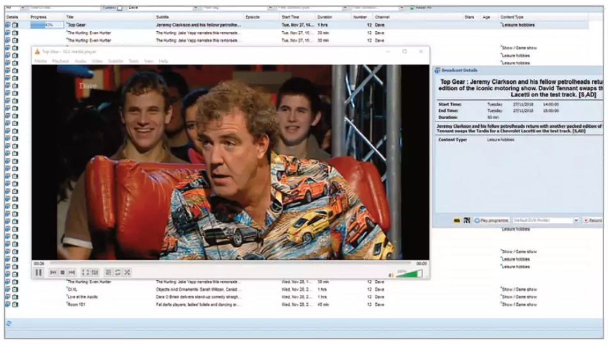
Notes :

- Many providers include undesirable services - Teleshopping, Adult Entertainment, etc; using the 'Map all services' will include these.
- You may need to enable specific EPG grabbers to receive OTA EPG data, See the EPG Grabber Modules Help doc for details.

Map all services:	✓
Create provider tags:	✓
Create network tags:	✓

BBC 6 Music DVB-T Network

STEP 10 The TVheadend server webpage will now display the list of available channels. After picking one, either click the title of the programme showing to expand the details, then click the **Play Programme** button to view its content, or click the small TV icon in the details column of the channel. You can now watch live TV across your home network.



Pi Projects: Desktop Pi

As you are now no doubt aware, the Raspberry Pi is certainly a versatile little piece of technology. Let's look at some of the popular projects you can apply to your Pi, starting with one that's not only the easiest, but also one of the best.

COMPUTER PI

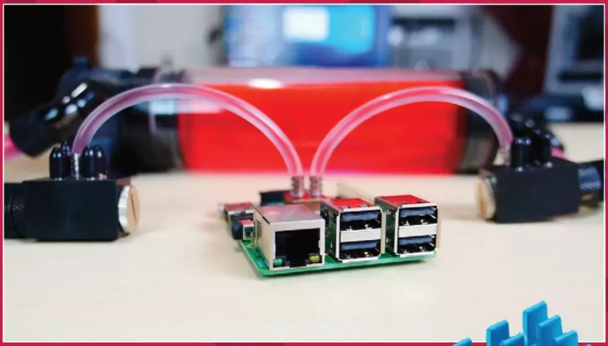
First and foremost, the Raspberry Pi is a computer. It has an operating system, built-in productivity apps, the ability to connect to both a home network and the Internet (either wirelessly or through Ethernet), and you can connect a mouse, keyboard, and monitor. It stands to reason, then, that it makes for a remarkably cheap desktop computer.

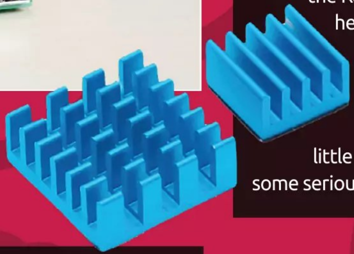
The obvious advantage of using the Raspberry Pi as a desktop computer though, is the fact that it's so cheap. For around £100 you could easily purchase the Pi, monitor, keyboard and mouse, providing you with a fully-functional computer capable of doing most, if not all, of the things you would do on a computer that costs ten times that amount. However, there are ways in which you can further enhance the desktop Pi project.

CASES

To begin with, you will need the Raspbian Stretch with desktop and recommended software OS version from the Raspberry Pi Foundation page, **www.raspberrypi.org/downloads/raspbian**. Once you've installed that to your Pi, booted it, and logged in with a personal user account, you can then start looking at one of the many colourful, and functional, cases available for the Pi.

The Pi Hut offers a superb selection of cases for the various versions of the Raspberry Pi, for this project, we'd recommend the Raspberry Pi 3 Model B+, as it's the fastest and most capable of the current models. For just £6, you can have a colourful – white and red – official Raspberry Pi case, with pre-formed holes that line up perfectly with the ports on the Pi. For around ten pound more, there's also the FLIRC case, an impressive design that's finished in brushed aluminium. On the other hand, if you want something a little different from the norm, then how about a case that's designed to look like a Nintendo Entertainment System, a SNES, or even a SEGA Megadrive?





COOLING

Another thing to consider, if you're going to be using your Pi as a desktop computer, is cooling. While the Pi's low use of resources means that its internals don't get too hot, prolonged and intense use may see your Pi getting a little warm, especially if it's inside a case and the ambient temperature of the environment is also warm. There are options available to help cool your Pi down, chiefly the Raspberry Pi Heatsink. This is a high-quality aluminium heat dispersing element that fits on top of the Pi's processor, leeching heat away from the vital circuit board and out into the surrounding air. However, bigger and more effective heatsinks are available, as well as fans, and even a water-cooling kit. These are a little extreme though, and only necessary if your Pi is doing some serious number crunching for an extended period of time.

STORAGE

While the Raspberry Pi desktop project is a great idea, the sad fact of the matter is that the Pi doesn't exactly offer the user a wealth of storage; compared to a Windows or Mac desktop, that is. Naturally, you can use a larger capacity SD card on which to install Raspbian, and utilise the remaining space as your Home folder, but again you're limited. Thankfully, it's not too much expense to include an external hard drive. A portable, USB connected, slimline 2TB hard drive can cost in the region of £60 and give you as much storage capacity as one of the more traditional desktop computers. One thing worth noting with regards to using an external hard drive, make sure you always have a backup of what's on there.

There are many options available to you if you decide to go down the Raspberry Pi desktop computer project route; there's even a Pi laptop kit available called Pi-Top. Needless to say, with this amount of choice you're able to customise your desktop and its setup in a more personal way than on a Windows PC, or a Mac.

PERIPHERALS

To cut down on the amount of cable clutter on your desktop, you could consider opting for a Bluetooth keyboard and mouse. The Pi 3 Model B+ offers Bluetooth connectivity alongside its Wi-Fi capabilities, so effectively you're able to use any recently purchased Bluetooth keyboard and mouse kit available. The major advantage here is that the Pi can be mounted to the rear of your monitor and, together with connecting to your home network via Wi-Fi, the only cables involved would be the power and the HDMI cable to the monitor, both of which could be neatly tucked away.

DON'T THROW AWAY YOUR PC YET

While the Raspberry Pi is a great little desktop computer, it does have its limitations. If you're accustomed to using a reasonably powerful Windows PC, or Mac, then you may find that, while being efficient, cheap and offering more space on your desk, the Pi does tend to struggle from time to time.

This is purely down to the fact that the Raspberry Pi isn't the most powerful computing device available today. It won't be able to handle the latest, triple-A games, intense graphics, Virtual Reality, or even some of the higher definition media content. You may find it stuttering when trying to playback fast moving 1080HD scenes, and true 4K playback isn't an option.

However, if you're looking to substitute your day-to-day workstation with something cheaper and smaller, and you're not going to push the Pi's processor too much, then you'll find the Pi to be a wonderful desktop computer. If you want more, though, hang on to that more powerful PC for the moment.

Pi Projects: Retro Gaming

Those of you old enough to recall the golden era of the home computer, the 80s, will have fond memories of playing on a Commodore 64, ZX Spectrum, Amiga, Atari ST, and the countless wonderful arcade machines that consumed more than their fair share of our pocket money. Hold on, because you're in luck.

GOING RETRO

The Raspberry Pi makes for an amazing retro gaming computer. With it you're able to emulate and play the classic home computers, consoles and arcade machines that brought us so much joy back in the day.

Thankfully, the processing power of most of the systems of the past is well within the capabilities of the Raspberry Pi's processor. There are a few examples that don't run too well, such as a PS2, or those systems that utilised a specialised 3D component, but on the whole, there's likely to be a fully working emulator, available for the Pi, that covers the home computer, console, or arcade that you remember playing.

RetroPie is the foremost retro gaming project available for the Raspberry Pi. It's a set of modules that are built upon Raspbian, an older project called EmulationStation, and Linus distribution called RetroArch. It contains built-in emulation that covers dozens of systems, from an Amiga through to an Atari 2600, Amstrad CPC to SEGA Dreamcast, ZX Spectrum to an Apple II; and the list is continually growing thanks to the contributions from the community.

RetroPie can work as an installation on top of an existing operating system, such as Raspbian, or you can install RetroPie to an SD card and boot the Pi directly into it, choosing to add further software later if you want. Once installed, you're able to connect USB controllers, or even a PS4 controller via Bluetooth, and if you're feeling up to it, there's also support for original controllers when connected to the Raspberry Pi's GPIO pins.

If you don't want to go down the RetroPie route and instead you only want one or two systems emulated, then Raspbian has a wealth of individual emulators available for you to install. You can search for the system via Google, or if you already know the name of it, simply install it to Raspbian through the Terminal. It's worth mentioning that there will often be multiple emulators designed for the same system. While one may work perfectly with the vast majority of games, it may struggle with some of the better games available for that particular system. On the other

hand, another emulator may play all the games for a system almost perfectly, with perhaps a loss of sound in some parts, or a slow down in others. It's therefore up to you what you want. You can have multiple emulators installed for a single system and use them depending on which game you want to play, based on how well the emulator supports them, or you can find the single emulator that works reasonably well with everything. It's trial and error, finding the perfect setup.

There are further options you can pursue when building a Raspberry Pi retro gaming system. You can encase your Pi inside one of the many retro-themed cases, such as the Mega-Pi SEGA Megadrive case, or you can build your Raspberry Pi into the Picade desktop arcade machine.

Picade is a great project, featuring authentic sticks and buttons, and an external speaker. There's artwork available for the arcade cabinet setup, along with full instructions on how to set up the Pi and connect everything to the GPIO pins. The 8-inch display is perfect for old-school gaming, and you can improve on the visuals by adding extra stickers, a different acrylic marquee, and posters. This, combined with a RetroPie installation, is an excellent project idea that'll keep you entertained for hours.

ABOUT ROMs

A ROM is the actual contents of a game, or the BIOS of an old system, be that a home computer or a console. These ROMs are often ripped from the original tape, cartridge, or chip, and are available to download from the Internet.

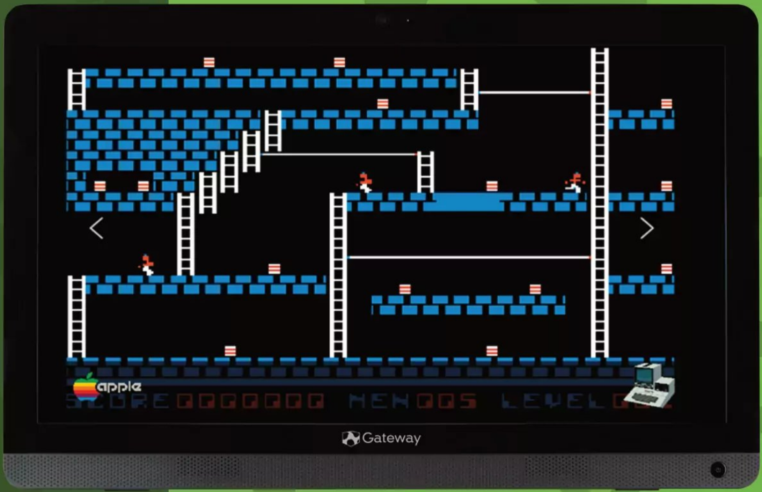
However, the use of ROMs is a continual legal headache. Most ROMs are illegal, meaning that they are available to download without the permission of the developer who created the game, the publishing house that released the game, and the company that owns the rights to the system on which the game is intended to be played.

There's a school of thought that if you own the original tape, cartridge and so on, of the game then you're legally allowed to obtain a ROM of the same game and play it on an emulator, but that's not always correct. In the example of music, it is technically illegal to copy music from a CD you own to playback on a media device; and in some ways the same applies to a ROM.

There are however, some examples of a situation when a title, or a system, has fallen out of copyright, or has been abandoned. In these instances, it is perfectly legal for someone to generate a ROM from the game and distribute it on the Internet. Sometimes, another developer improves the original game, adding extra levels, effects, and so on. Some original developers of older games have allowed the use of their game to be distributed as a ROM, and therefore it is legal for you to download and play it.

In short, if you download a ROM from the Internet, and it doesn't specify that it's abandonware or allowed by the developer, publishing house, or company, then you're doing so illegally. It's highly unlikely that the police will drop in through your bedroom window and arrest you on the spot – unless you actively host illegal content on a website you own – so it's purely down to your own conscious.

Remember, even if the game is over thirty years old, somewhere out there is a developer who spent the time creating it, so by playing a pirated version of it - an illegal ROM, you're taking something away from the individual or team that programmed the game in the first place.



LEGAL EMULATION

Interestingly, while it's illegal to download most ROMs, it is not illegal to install and use an emulator. In fact there are emulators available for the PS4 that allow you play old SEGA Megadrive games, so while Sony, or whoever released the emulator for the PS4, have paid for the rights to use the ROMs in their emulator, it isn't illegal for them to develop and use the SEGA Megadrive emulator.

In the same respect, it's not illegal for you to download an emulator for any of the systems, you will just need to find legal ROMs to play on it.

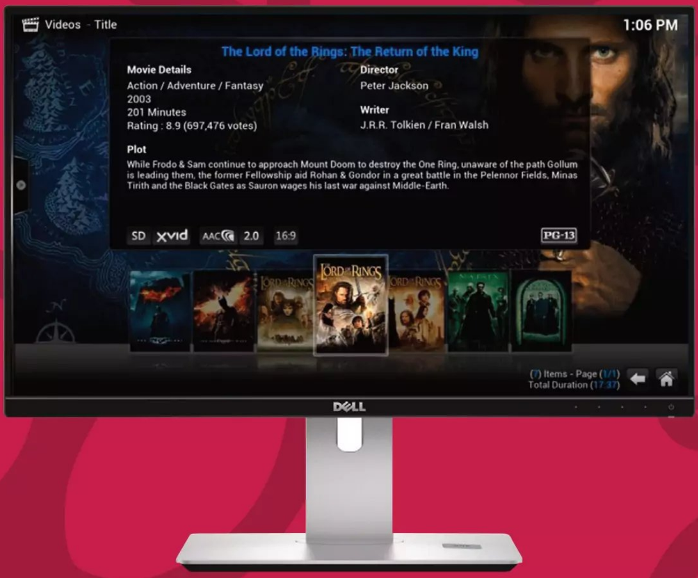
Pi Projects: Media Centre

Of all the Raspberry Pi projects out there, turning the tiny computer into a powerful media centre is probably top of the list. And thanks to the Pi's diminutive dimensions, and reasonable processing performance, it is wholly achievable.

FILM NIGHT

First off, what is a media centre? Basically, a media centre, with regards to what we're talking about here, is a computer that's capable of playing, possibly recording, and even sorting a media collection made up of videos, music and images. It's a home cinema system, connected to a TV that will stream media from online, or from a network attached storage device, or storage connected directly to the computer itself.

A media centre is often capable of sorting your collection of videos, music, and photos into a logical order, either by album, genre, year, alphabetically, or some custom setup. It can connect to the Internet and display information regarding the media, such as album cover, movie poster, location an image was taken, or even connect to an online database such as IMDB and provide further information. In short, it's a one-stop location for all your content.


Videos - Title 1:06 PM

The Lord of the Rings: The Return of the King

Movie Details
Action / Adventure / Fantasy
2003
201 Minutes
Rating : 8.9 (697,476 votes)

Director
Peter Jackson

Writer
J.R.R. Tolkien / Fran Walsh

Plot
While Frodo & Sam continue to approach Mount Doom to destroy the One Ring, unaware of the path Gollum is leading them, the former Fellowship aid Rohan & Gondor in a great battle in the Pelennor Fields, Minas Tirith and the Black Gates as Sauron wages his last war against Middle-Earth.

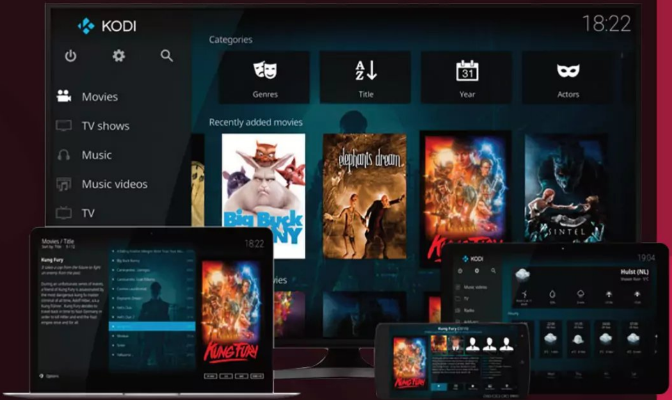
SD xvid AAC 2.0 16:9 PG-13

(7) Items - Page (1/1)
Total Duration (17:37)

KODI

Since the Pi's first appearance, there has been some form of media centre software available for it. Back then, when it launched in 2012, the major player was XBMC, Xbox Media Centre, and was considered one of the best cross-platform media centre applications ever developed. By 2014, though, the team behind XBMC decided to change its name to Kodi.

Kodi itself is an entertainment hub that collates and organises all your digital media into a single, user-friendly, and beautifully designed user interface. Thanks to its design, Kodi is capable of playing all types of music media and video media, as well as streaming TV shows (via online, or through a compatible TV tuner) and photos, plus you can record live TV through a PVR. It's extraordinarily customisable, allowing you to install further add-ons that can connect to other online services like SoundCloud, YouTube and so on.



AUTO-START KODI

If you want to start Kodi as soon as your Raspberry Pi powers up, then you will need to alter one of the configuration files. While in the Terminal, after installing Kodi, enter:

```
sudo nano /etc/default/kodi
```

Look for the entry ENABLED, and change it to 1:

```
ENABLED=1
```

Press Ctrl+Z to save and exit Nano, then reboot your Pi with:

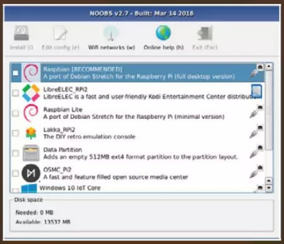
```
sudo reboot
```

Your Pi will now reboot and Kodi will start up once the main boot sequence has completed.

LIBREELEC VS. OSMC VS. RASPBIAN

Which OS to install? We could argue all day over the benefits, advantages, and disadvantages of each available operating system and you will probably still be as undecided as when we began.

In truth, there is no perfect operating system, as each offers its own unique way of doing things. In the end, it simply comes down to which one do you prefer. Raspbian has everything, yet will be slower, LibreELEC and OSMC are faster, but aren't as fully featured as Raspbian. What you need to do is experiment with all three, have a look around to see what else is available, and then decide on which works best for you and your particular goals with the Raspberry Pi.


NOOBS v2.7 - Built: Mar 14 2018

Raspbian [RECOMMENDED]
A port of Debian Stretch for the Raspberry Pi (full desktop version)

LibreELEC_RPi2
LibreELEC is a fast and user-friendly Kodi Entertainment Center distribut

Raspbian Lite
A port of Debian Stretch for the Raspberry Pi (minimal version)

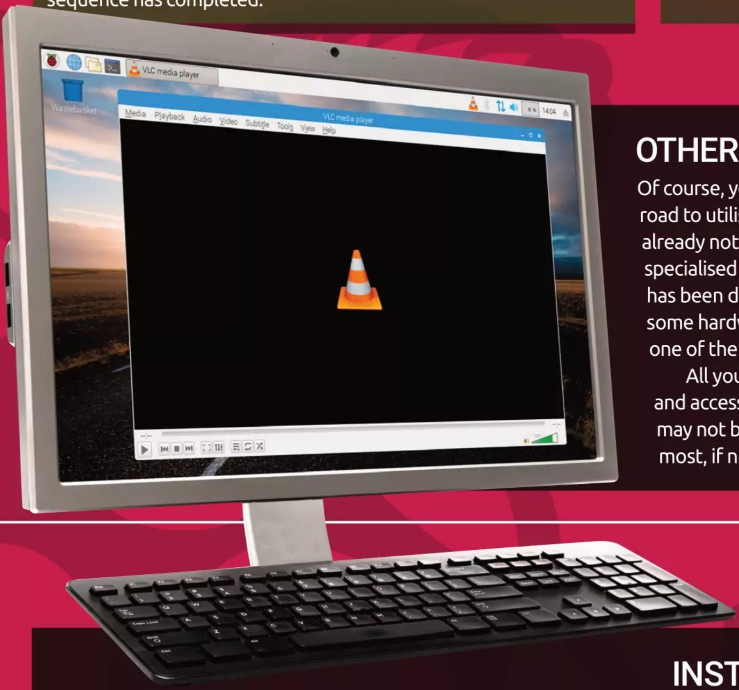
Lakka_RPi2
The DIY retro emulation console

Data Partition
Adds an empty 512MB ext4 format partition to the partition layout.

OSMC_Pi2
A fast and feature filled open source media center

Windows 10 IoT Core

Disk space
Needed: 0 MB
Available: 13537 MB



OTHER OPTIONS

Of course, you don't necessarily need to go down the full media centre road to utilise the Pi as an excellent media playback device. As you have already noticed, the Raspberry Pi comes pre-installed with its own specialised version of VLC, a very capable media player. VLC on the Pi has been designed to make use of the Pi's processing power, and has some hardware acceleration benefits coded directly into it, making it one of the best media players available.

All you need, therefore, is to have a software updated Pi with VLC, and access to wherever you've stored your media collection. While it may not be as graphically impressive as Kodi, you can still playback most, if not all, forms of digital media.

INSTALLING KODI – OPTION 1

Should you want to install Kodi on your Raspberry Pi and use it as an all-powerful media centre, there are two options available to you. The first is to simply boot up Raspbian, drop into a Terminal session, perform an update and upgrade, then enter:

```
sudo apt-get install kodi
```

Once the setup is complete, you can start Kodi by entering: kodi, into the Terminal.

INSTALLING KODI – OPTION 2

The second option for installing Kodi on the Pi is to use one of the Kodi-optimised operating systems suitable for the Raspberry Pi. If you recall, from when you first installed Raspbian via NOOBS, there are several OS options available to you of which two are LibreELEC and OSMC. Both of these operating systems are especially designed to work better with Kodi than the fully installed version of Raspbian, due to being a more lightweight OS and less system resource heavy.

This option is often regarded as the preferred method of turning the Pi into a media centre, as both LibreELEC and OSMC are much faster operating systems, giving more of the Pi's resources over to Kodi and allowing it to playback content without causing too many problems.

Pi Projects: BBS Client

In a digital world, before the Internet was a common household name, there existed a connected community of surfers. These individuals didn't surf the WWW, instead they dialled up Bulletin Board Systems (BBS) and opened a whole new world of content.

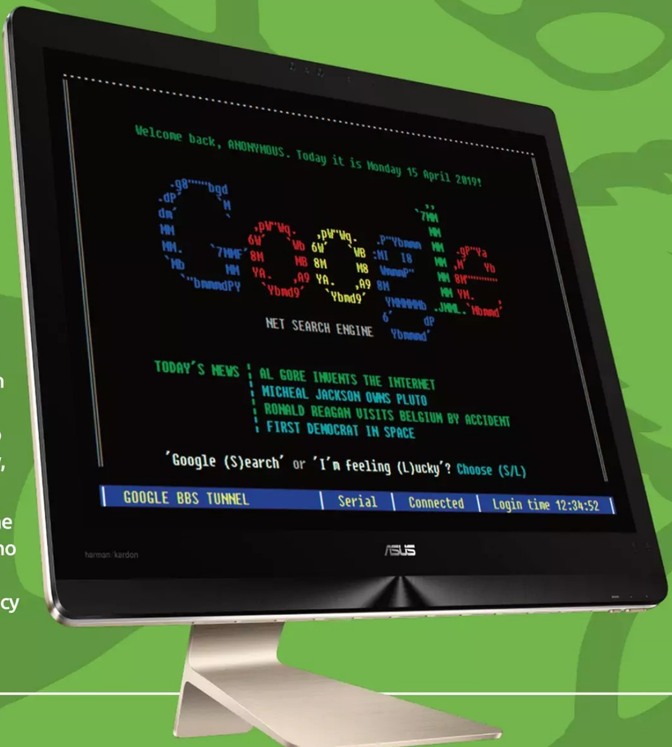
WARGAMES

If you're old enough to recall, or have since watched, the excellent movie Wargames, then you'll be roughly familiar with the way in which a Bulletin Board System works; and if you haven't watched Wargames, then we recommend you get hold of it.

In the movie, the young protagonist spends his days at the keyboard of his early 80s computer, using his modem to dial into remote systems. Once inside these remote systems, he then goes about traversing the remote host's file system looking for anything interesting.

The movie plot aside, this, essentially, is how a BBS works. It's a remote computer that runs specialist BBS server software with a mix of content either pre-installed, or added by the system admin (sysadmin or sysop). A user of the BBS can then dial, in the old modem sense, the BBS server's phone number and gain access to the system with a valid username and password; or if they're new, they have the option to create a new user.

These days, of course, the dial-up aspect has pretty much gone the way of the Dodo (although there are still some retro stalwarts who relish in the chronic noise of a dial-up connection), however, we can still enjoy the retro feeling of a traditional BBS using the legacy protocol, Telnet.



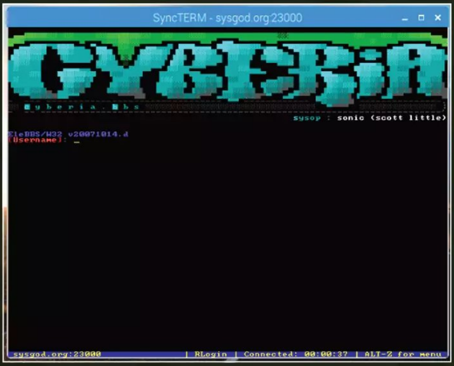
```
Welcome back, ANONYMOUS. Today it is Monday 15 April 2019!

NET SEARCH ENGINE

TODAY'S NEWS | AL GORE INVENTS THE INTERNET
             | MICHEAL JACKSON OWNS PLUTO
             | RONALD REAGAN VISITS BELGIUM BY ACCIDENT
             | FIRST DEMOCRAT IN SPACE

'Google (S)earch' or 'I'm feeling (L)ucky'? Choose (S/L)

| GOOGLE BBS TUNNEL | Serial | Connected | Login time 12:34:52 |
```

WHY?

In a world of Internet snooping, a BBS is probably one of the last bastions of digital privacy; to some degree. A private BBS is somewhere you can connect with likeminded individuals, to chat, swap code, reminisce, play a text-based adventure, or simply just hang out. True, you can get hold of copyrighted or explicit content, but that's only if you connect to those BBSes that serve such content, just as with the Internet.

Most BBSes follow a theme, whether that's old DOS-based adventures, ZX Spectrum fans, Commodore 64 gamers, or even something non-techie related, such as a Ford Cortina owners club; no doubt swapping owner manuals, old photos of the Cortina E and such.

In truth, a modern BBS is a bit of fun. Connecting to a system someone has installed and built, set around a particular theme, and designed with fantastic looking ANSI graphics, is a great pastime. It's a form of respect, in some ways, to acknowledge the work that's gone into creating the BBS by connecting to it and you also get to learn a little more about how protocols work, and how everything is connected.



```
SyncTERM - sysgod.org:23000

CYBERIA
cyberia.bbs
                                        sysop : sonic (scott little)
EleBBS/W32 v20071014.d
[Username]: _

sysgod.org:23000            | RLogin | Connected: 00:00:37 | ALT-Z for menu
```

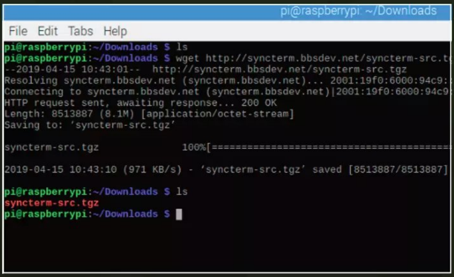
HOW?

Connecting to a BBS via the Raspberry Pi is quite easy, but to get the most from it you will need to get your hands dirty in the Terminal.

As mentioned, we're going to be using a form of the protocol Telnet, in this instance specifically the program SyncTERM, to emulate the old-style Terminals that support ANSI art and IBM fonts, while connecting to the remote BBS with the Telnet protocol. You can simply use telnet under the Terminal (once you've installed it), but you'll miss some of the glorious artwork displayed within the majority of the BBSes.

To begin with, drop into a Terminal session on your Pi. When the Terminal is fired up, enter sudo apt-get update && sudo apt-get upgrade to ensure your system is up to date. If everything is okay, enter: sudo apt-get install telnet. While this stage isn't strictly necessary, it's always a good idea to have the base protocol client installed.

When telnet is installed, you can then start the procedure of installing SyncTERM. In order to get SyncTERM working, you'll need to build it from source. By now, you should be a dab hand at this, but here's the process in case you've forgotten (along with some added elements to help everything go to plan).



```
pi@raspberrypi:~/Downloads $ ls
pi@raspberrypi:~/Downloads $ wget http://syncterm.bbsdev.net/syncterm-src.t
--2019-04-15 10:43:01--  http://syncterm.bbsdev.net/syncterm-src.tgz
Resolving syncterm.bbsdev.net (syncterm.bbsdev.net)... 2001:19f0:6000:94c9:
Connecting to syncterm.bbsdev.net (syncterm.bbsdev.net)|2001:19f0:6000:94c9:
HTTP request sent, awaiting response... 200 OK
Length: 8513887 (8.1M) [application/octet-stream]
Saving to: 'syncterm-src.tgz'

syncterm-src.tgz              100%[========================================

2019-04-15 10:43:10 (971 KB/s) - 'syncterm-src.tgz' saved [8513887/8513887]

pi@raspberrypi:~/Downloads $ ls
syncterm-src.tgz
pi@raspberrypi:~/Downloads $ 
```

Begin by changing directory to the Downloads folder and downloading the source code:

```
cd Downloads\
wget http://syncterm.bbsdev.net/syncterm-src.tgz
ls
```

With ls entered, you should see the newly downloaded tgz file. To unpack the downloaded file, enter:

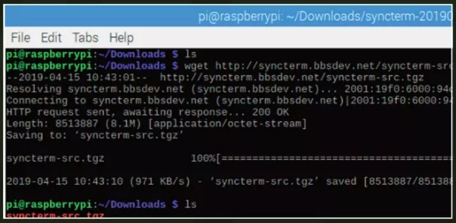
```
tar -xf syncterm-src.tgz
```

This will create a new syncterm-(DATE) folder, where DATE is the current date when you've unpacked the contents of the tgz file.

You will now need to change directories to:

```
cd syncterm-(DATE)
cd src
cd syncterm
```

You can mesh these directories together, but, for the sake of keeping things simple, we'll stick to one folder at a time. Also, remember you can hit the Tab key to auto-complete a directory name.



```
pi@raspberrypi:~/Downloads $ ls
pi@raspberrypi:~/Downloads $ wget http://syncterm.bbsdev.net/syncterm-sr
--2019-04-15 10:43:01--  http://syncterm.bbsdev.net/syncterm-src.tgz
Resolving syncterm.bbsdev.net (syncterm.bbsdev.net)... 2001:19f0:6000:94
Connecting to syncterm.bbsdev.net (syncterm.bbsdev.net)|2001:19f0:6000:9
HTTP request sent, awaiting response... 200 OK
Length: 8513887 (8.1M) [application/octet-stream]
Saving to: 'syncterm-src.tgz'

syncterm-src.tgz              100%[=====================================

2019-04-15 10:43:10 (971 KB/s) - 'syncterm-src.tgz' saved [8513887/85138

pi@raspberrypi:~/Downloads $ ls
syncterm-src.tgz
```

Once in the syncterm directory, you can begin to build from source. However, before you do that, it's worth installing a couple of extras to ensure the BBS session works to perfection. Start by installing the following:

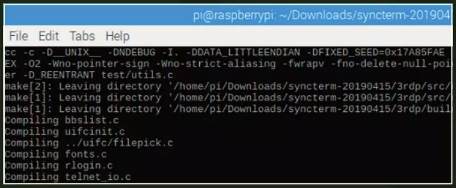
```
sudo apt-get install libncurses5-dev
sudo apt-get install libsdl1.2-dev
```

Once these two are installed, start the build process by entering:

```
sudo make
```

Followed by:

```
sudo make install
```



```
cc -c -D__UNIX__ -DNDEBUG -I. -DDATA_LITTLEENDIAN -DFIXED_SEED=0x17A85FAE
EX -O2 -Wno-pointer-sign -Wno-strict-aliasing -fwrapv -fno-delete-null-poi
er -D_REENTRANT test/utils.c
make[2]: Leaving directory '/home/pi/Downloads/syncterm-20190415/3rdp/src/
make[1]: Leaving directory '/home/pi/Downloads/syncterm-20190415/3rdp/src/
make[1]: Leaving directory '/home/pi/Downloads/syncterm-20190415/3rdp/buil
Compiling bbslist.c
Compiling uifcinit.c
Compiling ../uifc/filepick.c
Compiling fonts.c
Compiling rlogin.c
Compiling telnet_io.c
```

The process may take a few minutes, so be patient. When everything is installed, you can enter the command: syncterm, to start the program and change the screen settings, if you wish. However, to get straight into connecting to a BBS, try one of these commands:

```
syncterm dura-bbs.net: 6359
syncterm bbs.kernelerror.com: 10023
syncterm particlebbs.dyndns.org: 6400
syncterm heatwave.ddns.net: 9640
syncterm sysgod.org:23000
```

Naturally, some, or even all, of these BBSes may be offline when you come to test them; they are, after all, being operated by individuals like you and I. If they are offline, you can always get hold of a comprehensive list of active servers by visiting **https://www.telnetbbsguide.com/bbs/list/brief/**.

GET CONNECTED

It's worth spending some time finding the sort of BBS that suits your tastes. As you'll see by visiting the aforementioned website, there's over 500 BBSes currently listed, so somewhere in there a BBS could be your new online haunt.

Pi Projects: Weather Station

The Raspberry Pi has offered its userbase a superb platform to science. There are Raspberry Pi projects that involve the International Space Station, alongside NASA and ESA-led applications. However, it's an Earth-based project that's taken the community by storm.

IT'S RAINING PI, HALLELUJAH!

Weather stations have been available to the project-minded public since long before the Raspberry Pi entered the scene, but thanks to the Pi's unique specifications, a new generation of weather station has emerged; and they're really quite remarkable.

As with a lot of Raspberry Pi projects, the market has become flooded with various hardware, kits, and HATs that will enable you to set up a simple, or complex, weather station. The more experienced weather station and Raspberry Pi user will probably lean toward buying in the components individually, as opposed to an entire kit, but for the rest of us the kit-form versions are an excellent start to a project that can quickly grow over time.

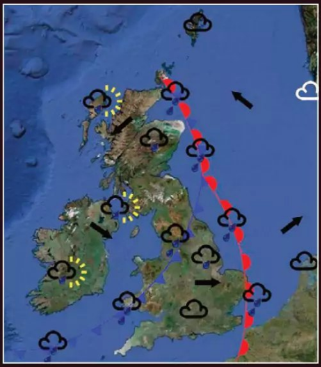
WHY?

With our smart devices connected to everything around us all the time, having a multi-limbed weather station, perched precariously on the end of the garden shed, measuring the amount of rainfall and in what direction the wind is blowing, may seem a little unnecessary; since your phone could tell you all that already with just a swipe or two. However, that discounts the pleasure of creating your own project and measuring the results as they change from one day to the next.

Any project, regardless of whether it's on a Raspberry Pi or not, is something from which to learn. In the case of a Raspberry Pi driven weather station, the user can learn about all manner of electronics, connectivity, some simple engineering, and coding, even before we get onto the varying aspects of meteorology.

The weather station project doesn't necessarily need to be an all-powerful outdoor contraption either. Basically, there are two routes you can opt to take: the outdoor model, measuring wind speed/direction, precipitation and so on, and an indoor model that measures ambient temperature, pressure and humidity.

Why indoors? There are a number of reasons. Perhaps you're in charge of a collection of servers, and while the company you work for may not be able to stretch to a fully sealed environment machine room, you could fill the gap with a great collection of sensors attached to a Raspberry Pi. You may have some from of allergy, where precise control of your indoor environment is important. Maybe you look after some exotic animals, in a room a where air pressure, humidity and such are vital to the creature's health. Needless to say, there are plenty more examples.



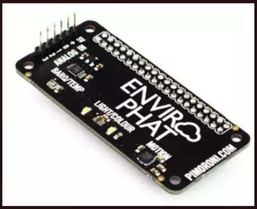
In short, setting up a weather station is simply a fun project. Setting aside the things you'll learn from getting one up and running, you'll eventually have a physical piece of hardware that can, with moderate accuracy, measure your environment and display that data in a number of graphs and charts. It's quite cool, when you stop to think about it.

SOME OPTIONS?

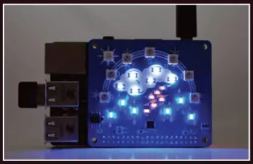
To go into all the currently available Raspberry Pi weather station options will take more space than we're currently able to offer here, but let's have a quick look at some of the best selling kits and components.

Weather Meter Kit – For outdoor weather monitoring, we have the Weather Meter Kit as sold by CPC for around £77. The kit comprises of three main component sensors that measure wind speed, wind direction, and rainfall. The rain gauge is a self-emptying bucket that's capable of activating for each 0.011-inch of rainfall collected, while the anemometer activates a closure switch every 1.492mph once per second. The wind direction module can display up to sixteen different positions, and the kit comes complete with full assembly instructions.

Velleman WS3080 Weather Station – The Velleman WS3080 Weather Station is one of the higher-end products that can interact with the Raspberry Pi. This outdoor mounted station is capable of monitoring rainfall, wind speed and wind direction, air pressure, wind chill temperatures, air temperature, and UV light levels. It features alarms for temperature, humidity, wind chill, dew point, wind speed, air pressure and storms, and comes with a separate LCD transmitter.



Pimoroni Enviro Phat – The Enviro pHAT is an add-on sensor for the Raspberry Pi that includes four different sensors to monitor temperature, pressure, light levels, and motion. It's ideal for indoor environment monitoring and, thanks to some nifty Python libraries and code, you're able to display the data via a web page and view the variables remotely.

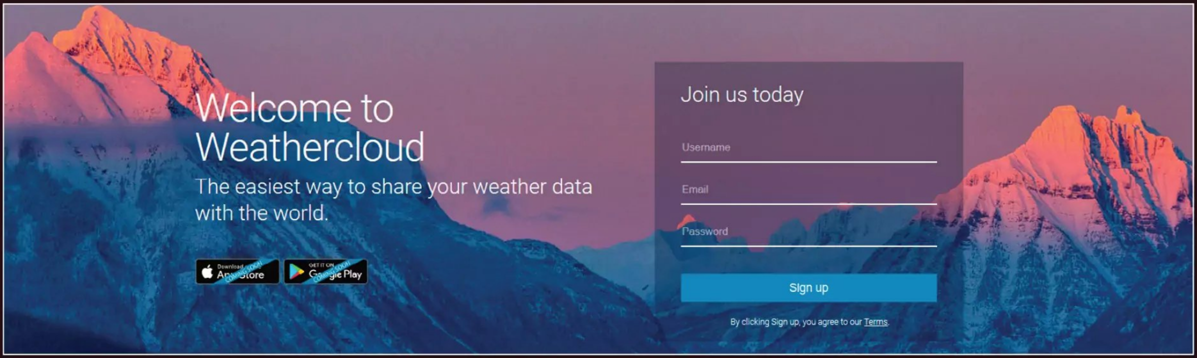


Cyntech WeatherHAT – The WeatherHAT from Cyntech can turn the Raspberry Pi into a weather display station. While not obtaining data directly from an external source (instead using one of the many localised, weather monitoring stations), it does offer seven RGB LEDs to highlight the current and upcoming weather. Controlled via Python, the HAT can help provide a good coding base for future weather station projects.

WeeWX – When you collect the necessary hardware together (the various sensors you'll need and so on), you'll need a good piece of software to help collate all that data and display it in a readable fashion; this is where WeeWX comes in. WeeWX is a Python-based program that can interact with many different weather stations to produce graphs and reports, and even publish the data to HTML on a web site.

WHERE NEXT?

As you can imagine, it'll take some time to get everything together, program the components, and collate the data you want to view. Once all that's gathered, though, what next?


Welcome to Weathercloud
The easiest way to share your weather data with the world.
App Store
Google Play
Join us today
Username
Email
Password
Sign up
By clicking Sign up, you agree to our Terms.

There are some websites available that display localised weather from independent sources such as a back garden weather station, so when you've got everything up and running, try looking out for such projects and start collaborating. Also, consider creating a Twitter account for the Raspberry Pi weather station and Tweeting the local weather.

Common Raspberry Pi Problems

The Raspberry Pi hardware and software is pretty reliable and problems are more often due to set up errors rather than anything to do with the hardware. However, there are times when hardware can seem to be at fault, so here are a few of the more common issues you might encounter when using your Raspberry Pi.

TROUBLESHOOTING YOUR RPI

The Raspberry Pi is a surprisingly stable bit of kit, but there is always the chance of encountering problems. If you're really stuck there are apps available which can help diagnose problems on the RPi.

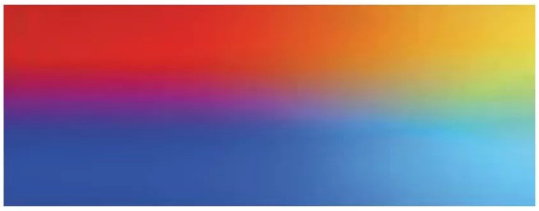
RED POWER LED IS BLINKING

A blinking red power LED indicates problems with the power supply. On model A and B, it is hard-wired to the 3.3V power supply rail. If it is blinking, it means the 5V power supply is dropping out. Use a different power supply. On the model B+ and also the A+, the circuit has been improved to give a much more reliable warning of poor power quality. The red power LED is wired to an APX803 supervisor which kicks in when the 5V power supply drops below 4.63V. If it does, the LED will blink. Check your connections, cable and power supply.

COLOURED SPLASH SCREEN

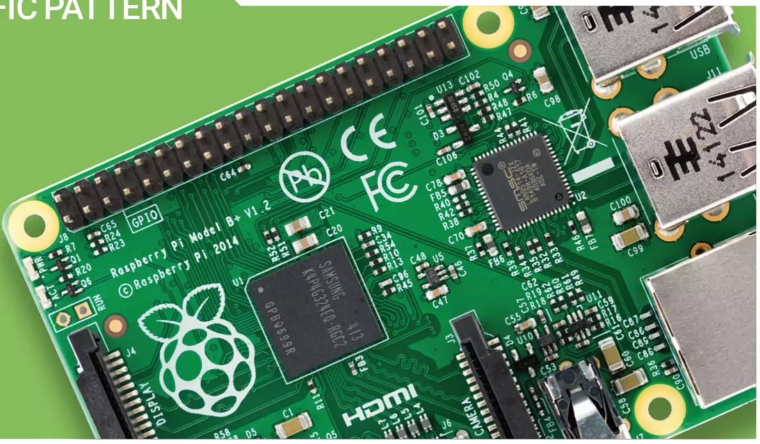
With the current firmware, a coloured splash screen is displayed after GPU firmware (start.elf) is loaded. This should be replaced by Linux console a second later. However if the coloured screen remains, it suggests the kernel.img file is failing to boot. Try replacing it with a known good one.

Immediately after displaying the splash screen, the Pi starts consuming a little more current. If the Pi resets at that moment, it is an indication that the power supply isn't able to deliver the full current that your Pi requires but dips its output voltage below a minimum when loaded with the full current the Pi needs.



GREEN LED BLINKS IN A SPECIFIC PATTERN

1 flash:	Possibly you have a RPi from Micron. Take a good look at the processor if it says M with an orbit round it. Using the latest software will solve your problem; also make sure you have a 4Gb SD card, as a 2Gb doesn't work.
2 flashes:	The SD Card cannot be read. A solution could be to forma the card and flash Raspbian with Pi Installer from Terminal.
3 flashes:	Start.elf not found.
4 flashes:	Start.elf not launched.
7 flashes:	Kernel.img not found.
8 flashes:	SDRAM not recognised. You need newer bootcode.bin/start.elf firmware.



NO USB DEVICE WORKS

The most common cause of USB devices working is low power supply voltage from a bad PSU, cable or USB hub; but it could also be that no clock signal is present. Return the board for a replacement if you think this is the case but before coming to this conclusion, confirm known good peripherals. A significant number of USB keyboards are not compatible with Raspberry so make sure you are using one made for Pi.

RASPBERRY PI NOT RESPONDING TO KEY PRESSES

This is most often caused by inadequate power. Use a good power supply and a good power cable. Some cheap cables that work with a mobile phone, cannot fully power the Pi. Some USB devices require a lot of power; most will have a label showing the voltage and mA requirements. Each one should be 5v 100mA max. Any more than this and they must be used with a powered USB hub. Try unplugging every USB device except the keyboard. You should also note that some keyboards have built in hubs and can try to draw 150mA; Pi can only handle 100mA per USB slot without a hub. Use the latest software too.

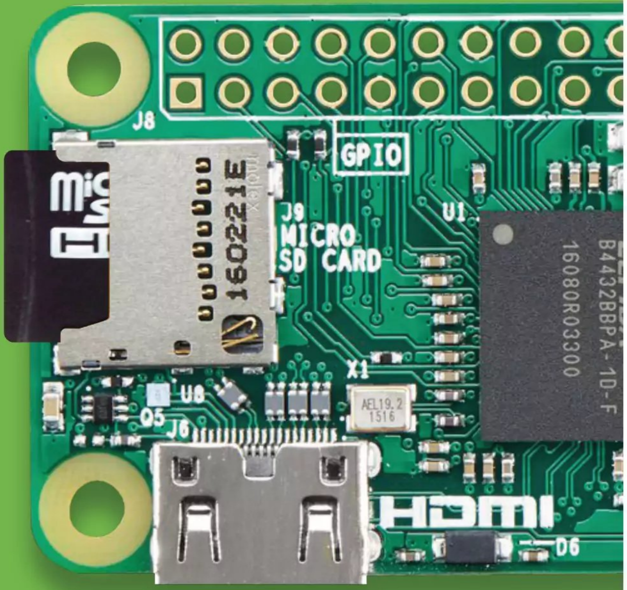
KEYBOARD OR MOUSE INTERFERES WITH A USB WI-FI DEVICE

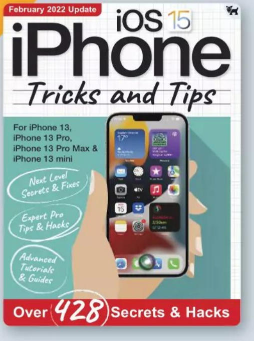
Connecting a keyboard and or mouse while a USB Wi-Fi device is connected, may cause one or both devices to malfunction. Tests point to interferences in the 2.4 GHz frequency band in which both Wi-Fi sticks, as well as USB keyboards transmit data. Changing the channel on the wireless access point should fix the problem completely.

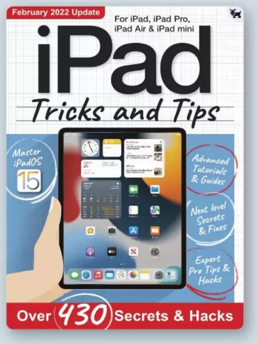
SD CARD PROBLEMS

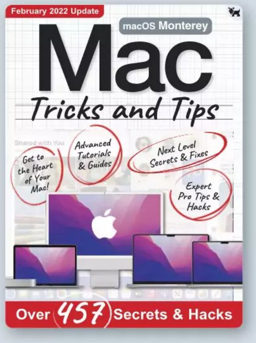
If you have problems, check you have the latest firmware version first. If that is not the problem, try the following.

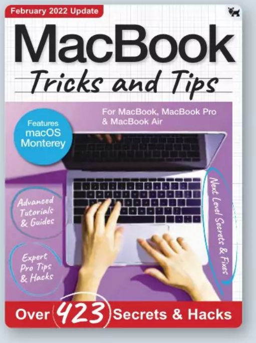
- Some SD cards do not work on the Pi, so check the list of known SD cards on the official Pi website.
- If you are having problems setting up your SD card you might want to start by erasing it completely, especially if it has been used elsewhere and still contains data or partitions.
- Windows and Mac users can download a formatting tool from the SD Association: **https://www.sdcard.org/downloads/formatter_3/**
- Reformatting cards is also easy to do in a digital camera.
- After writing the image to the SD card, verify that you can see the boot partition when you insert the SD card into your computer. The partition should contain a number of files, including start.elf and kernel.img. If you do not see these files on the SD card, you have made an error writing the image file.
- If you are manually preparing your SD card on Linux or macOS using the dd command, this operation will completely erase any existing data and partitions. Make sure you write to the whole card (e.g. /dev/sdd) and not to an existing partition (e.g. /dev/sdd1).
- If you put the SD card into your PC in an attempt to write the Pi operating system onto it and the PC tells you the card is write-protected, even with the write-protect tab in the correct forward position you may have a faulty SD-card rewriter.

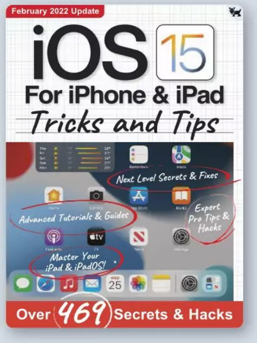


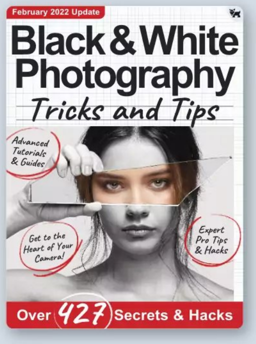

February 2022 Update
iOS 15 iPhone Tricks and Tips
Over 428 Secrets & Hacks

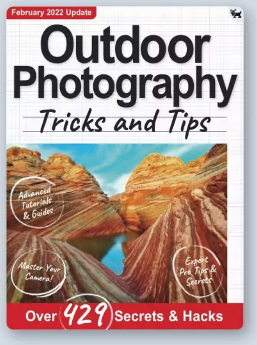

February 2022 Update
iPad Tricks and Tips
Over 430 Secrets & Hacks

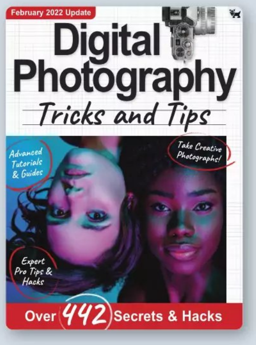

February 2022 Update
macOS Monterey Mac Tricks and Tips
Over 457 Secrets & Hacks

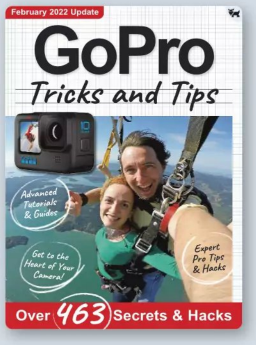

February 2022 Update
MacBook Tricks and Tips
Over 423 Secrets & Hacks

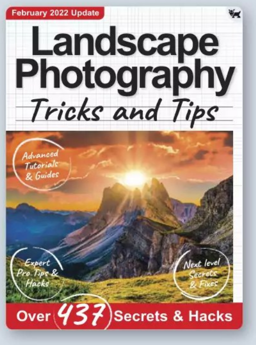

February 2022 Update
iOS 15 For iPhone & iPad Tricks and Tips
Over 469 Secrets & Hacks

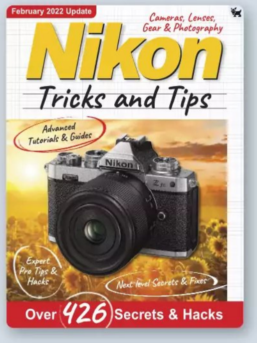

February 2022 Update
Black & White Photography Tricks and Tips
Over 427 Secrets & Hacks

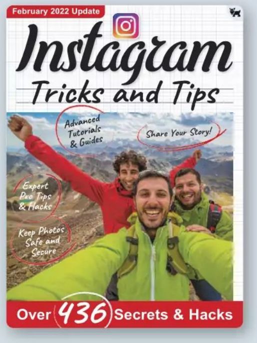

February 2022 Update
Outdoor Photography Tricks and Tips
Over 429 Secrets & Hacks


February 2022 Update
Digital Photography Tricks and Tips
Over 442 Secrets & Hacks

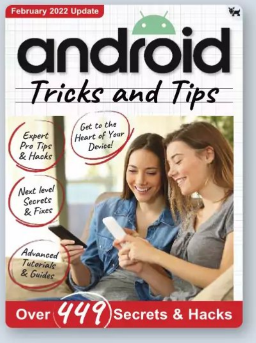

February 2022 Update
GoPro Tricks and Tips
Over 463 Secrets & Hacks


February 2022 Update
Landscape Photography Tricks and Tips
Over 437 Secrets & Hacks

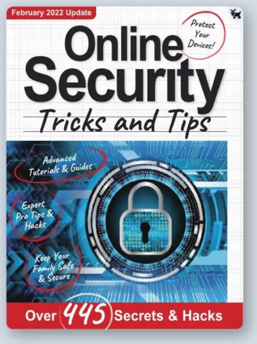

February 2022 Update
Nikon Tricks and Tips
Over 426 Secrets & Hacks

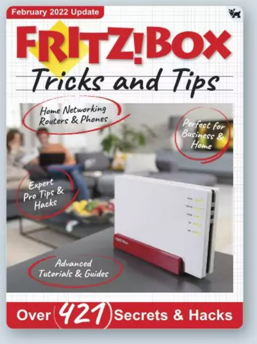

February 2022 Update
Instagram Tricks and Tips
Over 436 Secrets & Hacks

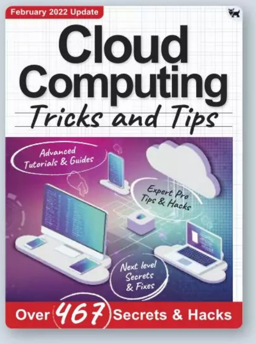
February 2022 Update
WhatsApp Tricks and Tips
Over 422 Secrets & Hacks

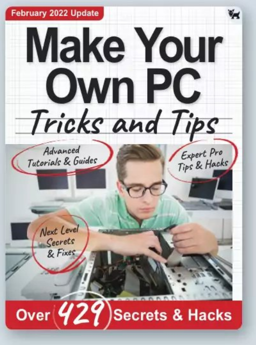

February 2022 Update
android Tricks and Tips
Over 449 Secrets & Hacks

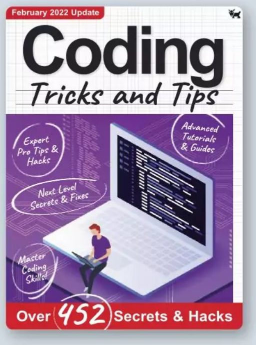
February 2022 Update
Samsung Galaxy Tricks and Tips
Over 432 Secrets & Hacks

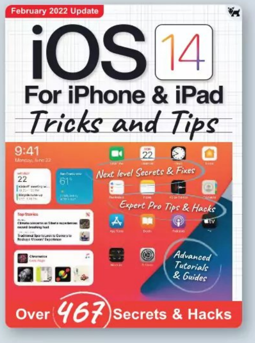

February 2022 Update
Online Security Tricks and Tips
Over 445 Secrets & Hacks

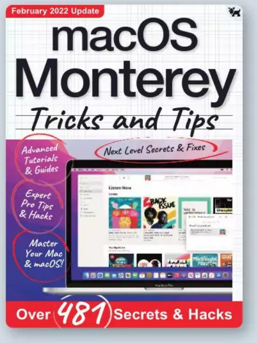

February 2022 Update
FRITZ!Box Tricks and Tips
Over 421 Secrets & Hacks

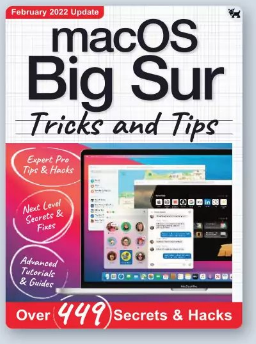

February 2022 Update
Cloud Computing Tricks and Tips
Over 467 Secrets & Hacks

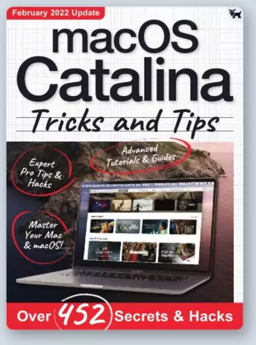

February 2022 Update
Make Your Own PC Tricks and Tips
Over 429 Secrets & Hacks

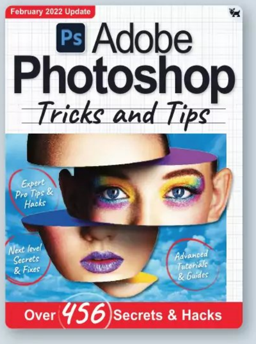

February 2022 Update
Coding Tricks and Tips
Over 452 Secrets & Hacks

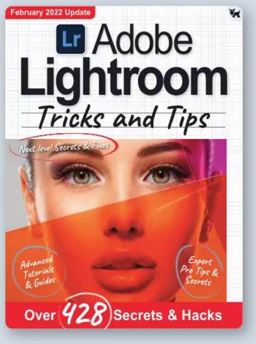

iOS 14 For iPhone & iPad
Tricks and Tips
Over 467 Secrets & Hacks

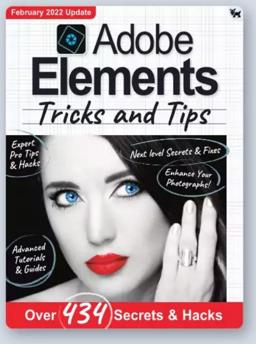

macOS Monterey
Tricks and Tips
Over 481 Secrets & Hacks

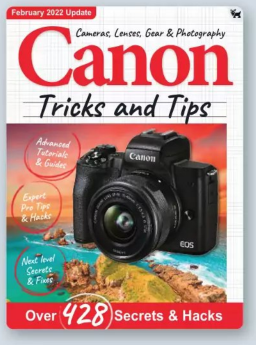

macOS Big Sur
Tricks and Tips
Over 449 Secrets & Hacks

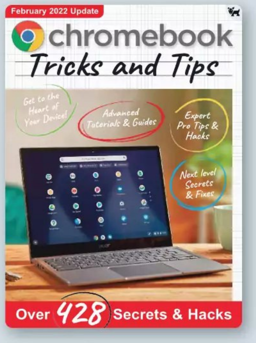

macOS Catalina
Tricks and Tips
Over 452 Secrets & Hacks

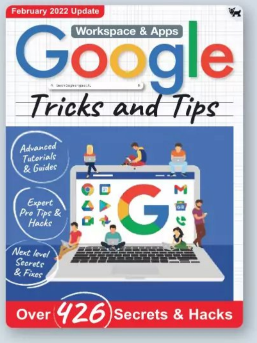

Adobe Photoshop
Tricks and Tips
Over 456 Secrets & Hacks

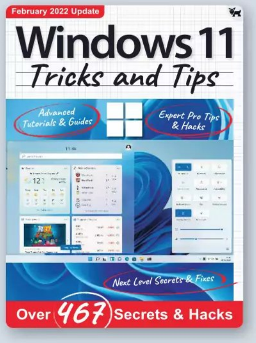

Adobe Lightroom
Tricks and Tips
Over 428 Secrets & Hacks

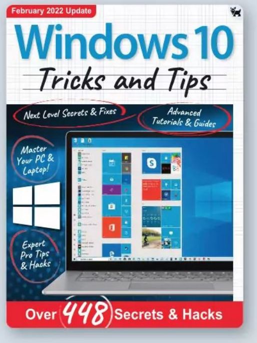

Adobe Elements
Tricks and Tips
Over 434 Secrets & Hacks

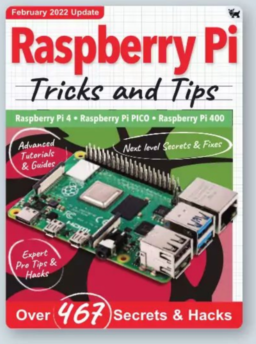

Canon
Tricks and Tips
Over 428 Secrets & Hacks

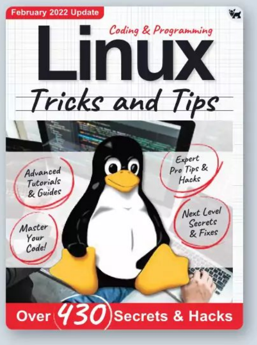

chromebook
Tricks and Tips
Over 428 Secrets & Hacks

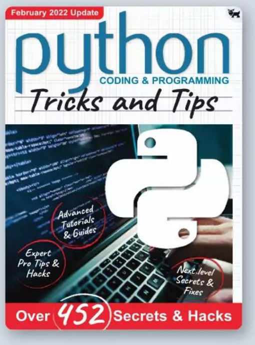

Google Workspace & Apps
Tricks and Tips
Over 426 Secrets & Hacks

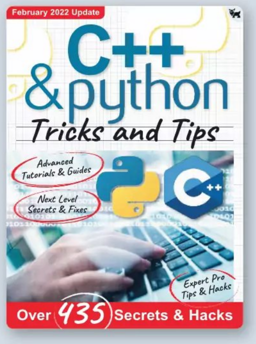

Windows 11
Tricks and Tips
Over 467 Secrets & Hacks


Windows 10
Tricks and Tips
Over 448 Secrets & Hacks


Raspberry Pi
Tricks and Tips
Over 467 Secrets & Hacks


Linux
Tricks and Tips
Over 430 Secrets & Hacks


python
Tricks and Tips
Over 452 Secrets & Hacks


C++ & python
Tricks and Tips
Over 435 Secrets & Hacks

Raspberry Pi Tricks and Tips
9th Edition | ISBN: 978-1-912847-70-9

Published by: Papercut Limited
Digital distribution by: Readly, Pocketmags & Zinio

Papercut Limited
Registered in England & Wales No: 04308513

INTERNATIONAL LICENSING
Papercut Limited has many great consumer tech, coding, photography and Photoshop image editing publications and all are available for licensing worldwide.
For more information email: james@bdmpublications.com