Tech for fun!

Release 1

Tom

April 24, 2015

CHAPTER

ONE

KID'S ACTIVITY

1.1 Minecraft Pi Edition

1.1.1 Basic commands

W	move forward
S	move backward
A	move left
D	move right
E	show inventory of blocks
1-8	select items in the quick bar
Space / Ctrl + Space	jump (ascend in fly-mode)
Shift / Ctrl + Shift	sneak (descend in fly-mode)
ESC	pause / menu
left mouse	destroy blocks
right mouse	place blocks
double **Space**	fly / fall
Tab	release mouse

1.1.2 Available blocks in Minecraft Pi Edition

AIR	Block(0)
STONE	Block(1)
GRASS	Block(2)
DIRT	Block(3)
COBBLESTONE	Block(4)
WOOD_PLANKS	Block(5)
SAPLING	Block(6)
BEDROCK	Block(7)
WATER_FLOWING	Block(8)
WATER	WATER_FLOWING
WATER_STATIONARY	Block(9)
LAVA_FLOWING	Block(10)
LAVA	LAVA_FLOWING
LAVA_STATIONARY	Block(11)
SAND	Block(12)
GRAVEL	Block(13)

Continued on next page

Table 1.1 – continued from previous page

GOLD_ORE	Block(14)
IRON_ORE	Block(15)
COAL_ORE	Block(16)
WOOD	Block(17)
LEAVES	Block(18)
GLASS	Block(20)
LAPIS_LAZULI_ORE	Block(21)
LAPIS_LAZULI_BLOCK	Block(22)
SANDSTONE	Block(24)
BED	Block(26)
COBWEB	Block(30)
GRASS_TALL	Block(31)
WOOL	Block(35)
FLOWER_YELLOW	Block(37)
FLOWER_CYAN	Block(38)
MUSHROOM_BROWN	Block(39)
MUSHROOM_RED	Block(40)
GOLD_BLOCK	Block(41)
IRON_BLOCK	Block(42)
STONE_SLAB_DOUBLE	Block(43)
STONE_SLAB	Block(44)
BRICK_BLOCK	Block(45)
TNT	Block(46)
BOOKSHELF	Block(47)
MOSS_STONE	Block(48)
OBSIDIAN	Block(49)
TORCH	Block(50)
FIRE	Block(51)
STAIRS_WOOD	Block(53)
CHEST	Block(54)
DIAMOND_ORE	Block(56)
DIAMOND_BLOCK	Block(57)
CRAFTING_TABLE	Block(58)
FARMLAND	Block(60)
FURNACE_INACTIVE	Block(61)
FURNACE_ACTIVE	Block(62)
DOOR_WOOD	Block(64)
LADDER	Block(65)
STAIRS_COBBLESTONE	Block(67)
DOOR_IRON	Block(71)
REDSTONE_ORE	Block(73)
SNOW	Block(78)
ICE	Block(79)
SNOW_BLOCK	Block(80)
CACTUS	Block(81)
CLAY	Block(82)
SUGAR_CANE	Block(83)
FENCE	Block(85)
GLOWSTONE_BLOCK	Block(89)
BEDROCK_INVISIBLE	Block(95)

Continued on next page

Table 1.1 – continued from previous page

STONE_BRICK	Block(98)
GLASS_PANE	Block(102)
MELON	Block(103)
FENCE_GATE	Block(107)
GLOWING_OBSIDIAN	Block(246)
NETHER_REACTOR_CORE	Block(247)

1.1.3 List of python programs

Short-cuts

Ctrl + S	save
F5	run

Display the player's position

```
from mcpi import minecraft

mc = minecraft.Minecraft.create()

x,y,z = mc.player.getTilePos()
mc.postToChat("x="+str(x)+", y="+str(y)+", z="+str(z))
```

Teleport (change the player's position)

In the following program, the player will be teleported 100 higher.

```
from mcpi import minecraft

mc = minecraft.Minecraft.create()

x,y,z = mc.player.getTilePos()
mc.player.setPos(x,y+100,z)
```

Build a huge block of activated TNTs

When you click one TNT, there will be an explosion around that block of TNTs.

```
from mcpi import minecraft

mc = minecraft.Minecraft.create()

x,y,z = mc.player.getTilePos()

tnt = 46
activated = 1
mc.setBlocks(x+1,y+1,z+1,x+5,y+5,z+5,tnt,activated)
```

Put a flower on the path

We will leave a flower when we are on a block of grass. Otherwise we will change the beneath block to a grass block.

```
from mcpi import minecraft
from time import sleep

mc = minecraft.Minecraft.create()

grass = 2
flower = 38
while True:
    x,y,z = mc.player.getTilePos()
    block_beneath = mc.getBlock(x,y-1,z)
    if block_beneath == grass:
        mc.setBlock(x,y,z,flower)
    else:
        mc.setBlock(x,y-1,z,grass)
    sleep(0.1)
```

Clear space with input size

We will clear space for a given **size**. To do so, we will build a cube of **size** x **size** x **size** blocks, filled with the AIR block.

```
from mcpi import minecraft, block

mc = minecraft.Minecraft.create()

x,y,z = mc.player.getTilePos()
size = int(raw_input("size of area to clear? "))
if size > 0:
    mc.setBlocks(x,y,z,x+size,y+size,z+size,block.AIR.id)
```

Challenge

Change a little the above program so that the player is in the middle of the cleared space (and also dig down a few blocks).

Build a house

```
from mcpi import minecraft, block

mc = minecraft.Minecraft.create()
SIZE = 20

def house():
    midx = x + SIZE/2
    midy = y + SIZE/2
    mc.setBlocks(     x,        y,   z,   x+SIZE,   y+SIZE,   z+SIZE,block.COBBLESTONE.id)
    mc.setBlocks(   x+1,      y+1,z+1,x+SIZE-1,y+SIZE-1,z+SIZE-1,         block.AIR.id)
    mc.setBlocks(   x+1,      y+1,z+1,x+SIZE-1,     y+1,z+SIZE-1,        block.WOOL.id,7)
    # left window
    mc.setBlocks(   x+3,y+SIZE-3,   z,   midx-3,   midy+3,        z,       block.GLASS.id)
```

```
    # right window
    mc.setBlocks(midx+3,y+SIZE-3,  z,x+SIZE-3,  midy+3,       z,      block.GLASS.id)
    # door
    mc.setBlocks(midx-3,       y,  z,  midx+3,    midy,       z,        block.AIR.id)
    # roof
    mc.setBlocks(     x,y+SIZE+1,  z,  x+SIZE,y+SIZE+1,  z+SIZE,       block.SNOW.id)

x,y,z = mc.player.getTilePos()

# build a house
house()
```

Build a street

```
from mcpi import minecraft, block

mc = minecraft.Minecraft.create()
SIZE = 20

def house():
    midx = x + SIZE/2
    midy = y + SIZE/2
    mc.setBlocks(     x,       y,  z,  x+SIZE,  y+SIZE,  z+SIZE,block.COBBLESTONE.id)
    mc.setBlocks(   x+1,     y+1,z+1,x+SIZE-1,y+SIZE-1,z+SIZE-1,        block.AIR.id)
    mc.setBlocks(   x+1,     y+1,z+1,x+SIZE-1,     y+1,z+SIZE-1,       block.WOOL.id,7)
    # left window
    mc.setBlocks(   x+3,y+SIZE-3,  z,  midx-3,  midy+3,       z,      block.GLASS.id)
    # right window
    mc.setBlocks(midx+3,y+SIZE-3,  z,x+SIZE-3,  midy+3,       z,      block.GLASS.id)
    # door
    mc.setBlocks(midx-3,       y,  z,  midx+3,    midy,       z,        block.AIR.id)
    # roof
    mc.setBlocks(     x,y+SIZE+1,  z,  x+SIZE,y+SIZE+1,  z+SIZE,       block.SNOW.id)

x,y,z = mc.player.getTilePos()

# build a street
for h in range(5):
    house()
    x += SIZE
```

`range(5) = range(0,5,1)`

`range(5)` means the list of the first five integers starting from 0, i.e. 0, 1, 2, 3, 4.
`range(5)` is indeed a shortcut of `range(0,5,1)` which means the list of integers starting from 0 and less than 5, increased 1 per step.

Magic bridge

Put a glass bridge under the feet whenever you walk, making sure that the player will never falls into the sea or falls out of the sky.

```
from mcpi import minecraft, block
import time
```

```
3
4  mc = minecraft.Minecraft.create()
5
6  def buildBridge():
7      x,y,z = mc.player.getTilePos()
8      b = mc.getBlock(x,y-1,z) # y-1 means under the feet
9      if (b == block.AIR.id) or
10        (b == block.WATER_STATIONARY.id) or
11        (b == block.WATER_FLOWING.id):
12         mc.setBlock(x,y-1,z,block.GLASS.id)
13
14 while True:
15     time.sleep(0.01)
16     buildBridge()
```

Vanishing bridge

Based on the previous magic bridge. Here we will add some more magic: whenever we are landed on a non-glass block, we will make disappear one glass block of our magic bridge.

To do this, we will build a list of all created glass blocks in order. When it will be the case, we will remove the oldest block in the list.

```
1  from mcpi import minecraft, block
2  import time
3
4  mc = minecraft.Minecraft.create()
5  bridge = []
6
7  def buildBridge():
8      x,y,z = mc.player.getTilePos()
9      b = mc.getBlock(x,y-1,z) # y-1 means under the feet
10     if (b == block.AIR.id) or
11        (b == block.WATER_STATIONARY.id) or
12        (b == block.WATER_FLOWING.id):
13         mc.setBlock(x,y-1,z,block.GLASS.id)
14         coordinate = [x,y-1,z]
15         bridge.append(coordinate)
16     elif b != block.GLASS.id:
17         if len(bridge) > 0:
18             coordinate = bridge.pop()
19             a,b,c = coordinate
20             mc.setBlock(a,b,c,block.AIR.id)
21             time.sleep(0.01)
22
23 while True:
24     time.sleep(0.01)
25     buildBridge()
```

Vanishing bridge (Improved Version)

One question raised by Tom on the previous version of vanishing bridge: can we remove also the bridge's one glass block when we are on a glass block which is not part of the bridge?

For this to be done, we need to check when we have a glass block, whether its coordinate is in the list of the bridge's glass blocks' coordinates.

```
from mcpi import minecraft, block
import time

mc = minecraft.Minecraft.create()
bridge = []

def buildBridge():
    x,y,z = mc.player.getTilePos()
    b = mc.getBlock(x,y-1,z) # y-1 means under the feet
    if (b == block.AIR.id) or
       (b == block.WATER_STATIONARY.id) or
       (b == block.WATER_FLOWING.id):
        mc.setBlock(x,y-1,z,block.GLASS.id)
        coordinate = [x,y-1,z]
        bridge.append(coordinate)
    elif b != block.GLASS.id:
        if len(bridge) > 0:
            coordinate = bridge.pop()
            a,b,c = coordinate
            mc.setBlock(a,b,c,block.AIR.id)
            time.sleep(0.01)
    else: # b == block.GLASS.id
        if [x,y-1,z] not in bridge:
            if len(bridge) > 0:
                coordinate = bridge.pop()
                a,b,c = coordinate
                mc.setBlock(a,b,c,block.AIR.id)
                time.sleep(0.01)

while True:
    time.sleep(0.01)
    buildBridge()
```

Vanishing bridge (Simplified Version)

There are two repeated blocks in the above code: it's our chance to create a new function!

```
from mcpi import minecraft, block
import time

mc = minecraft.Minecraft.create()
bridge = []

def popBridge():
    if len(bridge) > 0:
        coordinate = bridge.pop()
        a,b,c = coordinate
        mc.setBlock(a,b,c,block.AIR.id)
        time.sleep(0.01)

def buildBridge():
    x,y,z = mc.player.getTilePos()
    b = mc.getBlock(x,y-1,z) # y-1 means under the feet
    if (b == block.AIR.id) or
       (b == block.WATER_STATIONARY.id) or
       (b == block.WATER_FLOWING.id):
        mc.setBlock(x,y-1,z,block.GLASS.id)
```

```
        coordinate = [x,y-1,z]
        bridge.append(coordinate)
    elif b != block.GLASS.id:
        popBridge()
    else: # b == block.GLASS.id
        if [x,y-1,z] not in bridge:
            popBridge()

while True:
    time.sleep(0.01)
    buildBridge()
```

Treasure Hunt

```
from mcpi import minecraft, block
import time, random

mc = minecraft.Minecraft.create()

score = 0
RANGE = 5 # increase this number to make a more difficult game!
TIMEOUT = 10
timer = TIMEOUT

treasurex = None
treasurey = None
treasurez = None

def placeTreasure():
    global treasurex, treasurey, treasurez
    x,y,z = mc.player.getTilePos()
    treasurex = random.randint(x, x+RANGE)
    treasurey = random.randint(y+2, y+RANGE+2)
    treasurez = random.randint(z, z+RANGE)
    mc.setBlock(treasurex, treasurey, treasurez, block.DIAMOND_BLOCK.id)

def checkHit():
    global score, treasurex
    events = mc.events.pollBlockHits()
    for e in events:
        x,y,z = e.pos
        if x == treasurex and y == treasurey and z == treasurez:
            mc.postToChat("HIT!")
            score += 20
            mc.setBlock(treasurex, treasurey, treasurez, block.AIR.id)
            treasurex = None

def homingBeacon():
    global timer
    if treasurex != None:
        timer -= 1
        if timer == 0:
            timer = TIMEOUT
            x,y,z = mc.player.getTilePos()
            diffx = abs(x - treasurex)
            diffy = abs(y - treasurey)
            diffz = abs(z - treasurez)
```

```
            diff = diffx + diffy + diffz
            mc.postToChat("score:" + str(score) + " treasure:" + str(diff))

bridge = []

def buildBridge():
    global score
    x,y,z = mc.player.getTilePos()
    b = mc.getBlock(x, y-1, z)
    if treasurex == None:
        if len(bridge) > 0:
            coordinate = bridge.pop()
            a,b,c = coordinate
            mc.setBlock(a, b, c, block.AIR.id)
            mc.postToChat("bridge:" + str(len(bridge)))
            time.sleep(0.01)
    elif b != block.GOLD_BLOCK.id:
        mc.setBlock(x, y-1, z, block.GOLD_BLOCK.id)
        coordinate = [x, y-1, z]
        bridge.append(coordinate)
        score -= 1

while True:
    time.sleep(0.01)

    if treasurex == None and len(bridge) == 0:
        placeTreasure()

    checkHit()
    homingBeacon()
    buildBridge()
```

Build a Maze

```
from mcpi import minecraft, block

mc = minecraft.Minecraft.create()

GAP = block.AIR.id
WALL = block.GOLD_BLOCK.id
FLOOR = block.GRASS.id

FILENAME = "maze.csv"
f = open(FILENAME, "r")

x,y,z = mc.player.getTilePos()
ORIGINX = x+1
ORIGINY = y
ORIGINZ = z+1

z = ORIGINZ
for line in f.readlines():
    data = line.split(",")
    x = ORIGINX
    for cell in data:
        if cell == "0":
            b = GAP
```

```
        else:
            b = WALL
        mc.setBlock(x, ORIGINY, z, b)
        mc.setBlock(x, ORIGINY+1, z, b)
        mc.setBlock(x, ORIGINY-1, z, FLOOR)
        x += 1
    z += 1
```

maze.csv sample

```
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
1,1,1,1,1,1,1,1,1,0,1,0,1,1,0,1
1,0,0,1,0,0,0,0,1,0,1,0,1,0,0,1
1,1,0,1,0,1,1,0,0,0,0,0,1,0,1,1
1,1,0,1,0,1,1,1,1,1,1,1,1,0,1,1
1,1,0,0,0,1,1,1,1,1,0,0,0,0,1,1
1,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1
1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1
1,0,1,1,1,1,0,0,0,0,0,1,1,1,1,1
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
1,0,1,1,1,1,1,1,1,1,0,1,1,1,1,1
1,0,1,0,0,0,0,0,0,1,0,0,0,0,0,1
1,0,1,0,1,1,1,1,1,0,1,1,1,1,1,0,1
1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1
1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1
```

Challenge

- make your own CSV datasheet
- plant some random treasure

1.2 Pygame

1.2.1 List of pygame programs

Draw a circle

```
import pygame

width,height = 640,480
radius = 100
fill = 1

pygame.init()
window = pygame.display.set_mode((width,height))
window.fill(pygame.Color(255,255,255)) # white

while True:
    pygame.draw.circle(window,
                       pygame.Color(255,0,0), # red
```

```
                        (width/2,height/2),
                        radius,
                        fill)
    pygame.display.update()
    if pygame.QUIT in [e.type for e in pygame.event.get()]:
        break
```

Draw circles based on mouse move / position

```
import pygame
from pygame.locals import *

width,height = 640,640
radius = 0
fill = 1
mouseX,mouseY = 0,0

pygame.init()
window = pygame.display.set_mode((width,height))
window.fill(pygame.Color(255,255,255)) # white
fps = pygame.time.Clock() # FPS = Frame Per Second

while True:  # one frame per loop
    for event in pygame.event.get():
        if event.type == MOUSEMOTION:
            mouseX,mouseY = event.pos
        if event.type == MOUSEBUTTONDOWN: # mouse click
            window.fill(pygame.Color(255,255,255)) # clear screen
        radius = (abs(width/2 - mouseX) + abs(height/2 - mouseY))/2 + 1
        pygame.draw.circle(window,
                           pygame.Color(255,0,0), # red
                           (mouseX,mouseY),
                           radius,
                           fill)
    pygame.display.update()
    if pygame.QUIT in [e.type for e in pygame.event.get()]:
        break
    fps.tick(30) # wait so that frame rate is 30 fps
```

1.3 Scratch

CHAPTER

TWO

HARDWARE

2.1 Raspberry Pi

2.1.1 Default settings

login	**pi**
password	**raspberry**
hostname	**raspberrypi**
keyboard	UK

2.1.2 Basic commands

Config

```
$ sudo raspi-config
```

Start X server

```
$ startx
```

Reboot

```
$ sudo reboot
```

Shutdown

```
$ sudo shutdown -h now
```

Change datetime

```
$ sudo date --set="Sun Nov 18 1:55:16 EDT 2012"
```

Update

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

2.1.3 Information

Check OS version

```
$ cat /proc/version
```

Check board version

```
$ cat /proc/cpuinfo
```

Display network interface and associated IP addresses

```
$ ifconfig
```

2.1.4 Short-cuts

Ctrl + C	kill currently running program
Ctrl + D	exit shell
Ctrl + A	move cursor to the beginning of the line
Ctrl + E	move cursor to the end of the line
Ctrl + Alt + Backspace	[optional] terminate the X server

2.1.5 Setup Keyboard

The default keyboard is UK. Let's change it to AU keyboard.

The trick is that Australia is not listed in the country list for the keyboard, we need to setup a US keyboard instead.

Change the keyboard config

```
$ sudo vi /etc/default/keyboard
```

```
XKBMODEL ="pc105"
XKBLAYOUT="us"
XKBVARIANT=""
XKBOPTIONS=""

BACKSPACE="guess"
```

Then run the following commands and reboot

```
$ sudo setxkbmap -layout us
$ sudo udevadm trigger --subsysstem-match=input --action=change
```

2.1.6 Utilities / Softwares

raspi-config tool

```
$ sudo apt-get install raspi-config
```

Minecraft

```
$ sudo apt-get install minecraft-pi
```

Screenshot : scrot

```
$ sudo apt-get install scrot
```

Mercurial

```
$ sudo apt-get install mercurial
```

2.2 Arduino

CHAPTER

THREE

PROGRAMMING LANGUAGE

3.1 Shell

3.2 Python

CHAPTER

FOUR

EDITOR

4.1 Vi

4.1.1 Cursor Movement Commands

`(n)` indicates a number, and is optional

(n)h	left (n) space(s)
(n)j	down (n) space(s)
(n)k	up (n) space(s)
(n)l	right (n) space(s)

(The arrow keys usually work also)

CTRL + F	forward one screen
CTRL + B	back one screen
CTRL + D	down half screen
CTRL + U	up half screen
H	beginning of top line of screen
M	beginning of middle line of screen
L	beginning of last line of screen
G	beginning of last line of file
(n)G	move to beginning of line (n)
0	(zero) beginning of line
$	end of line
(n)w	forward (n) word(s)
(n)b	back (n) word(s)
e	end of word

4.1.2 Inserting Text

i	insert text before the cursor
a	append text after the cursor (does not overwrite other text)
I	insert text at the beginning of the line
A	append text to the end of the line
r	replace the character under the cursor with the next character typed
R	Overwrite characters until the end of the line (or until escape is pressed to change command)
o	(alpha o) open new line after the current line to type text
O	(alpha O) open new line before the current line to type text

4.1.3 Deleting Text

dd	deletes current line
(n)dd	deletes (n) line(s)
(n)dw	deletes (n) word(s)
D	deletes from cursor to end of line
x	deletes current character
(n)x	deletes (n) character(s)
X	deletes previous character

4.1.4 Change Commands

(n)cc	changes (n) characters on line(s) until end of the line (or until escape is pressed)
cw	changes characters of word until end of the word (or until escape is pressed)
(n)cw	changes characters of the next (n) words
c$	changes text to the end of the line
ct(x)	changes text to the letter (x)
C	changes remaining text on the current line (until stopped by escape key)
~	changes the case of the current character
J	joins the current line and the next line
u	undo the last command just done on this line
.	repeats last change
s	substitutes text for current character
S	substitutes text for current line
:s	substitutes new word(s) for old `:<line nos effected> s/old/new/g`
&	repeats last substitution (:s) command
(n)yy	yanks (n) lines to buffer
y(n)w	yanks (n) words to buffer
p	puts yanked or deleted text after cursor
P	puts yanked or deleted text before cursor

4.1.5 File Manipulation

:w (file)	writes changes to file (default is current file)
:wq	writes changes to current file and quits edit session
:w! (file)	overwrites file (default is current file)
:q	quits edit session w/no changes made
:q!	quits edit session and discards changes
:n	edits next file in argument list
:f (name)	changes name of current file to (name)
:r (file)	reads contents of file into current edit at the current cursor position (insert a file)
:!(command)	shell escape
:r!(command)	inserts result of shell command at cursor position
ZZ	write changes to current file and exit

4.2 VIM (Vi IMproved)

4.2.1 Basic commands

Read only (use :wq! to force the modification)

```
$ vim -R file
```

Running shell commands

```
!command
```

e.g. **!ls** will launch **ls**

if you wants to go directly to shell without quitting from VI editor you can go by executing **!sh / !bash / !ksh** from VI and then come back to VI editor by just executing command **exit** from shell. for Cygwin, **!bash** and **exit** seems to be the best choice

Launch VIM from command line

```
$ vi file.txt                       open and edit file file.txt
$ vi file1.txt file2.txt file3.txt  open several files
$ vi +25 file.txt                   edit from the 25th line
$ vi + file.txt                     edit at the end of file
$ vi +/text file.txt                edit from the first line containing the word test
$ vi -r file.txt                    restore a crashed file
$ view file.txt                     vi in read-only mode
$ vimtutor                          VIM tutorial
```

Saving and quiting commands

```
:w            save the current file (before quit)
:w file.txt   save the modified file with another file name
              (even if the file was opened in read-only mode)
:wq           save and quit
ZZ            save and quit
:q!           quit without saving
:wq!          save change in the current file opened in read-only mode, and then quit
:w!           save change in the current file opened in read-only mode
```

Checking history and help

```
:history        vim commands history
:help           all helps
:help command   help on one command
```

Recording and replaying commands

Recoding in vim or VI editor can be done by using **q** and the executing recorded comment by using **q@1**

4.2.2 Options

Here are the major VIM editor options

```
:set nu             This will display line number in front of each line quite useful if
                    you want line by line information. You can turn it off by executing
                    "set nonu". Remember for turning it off put "no" in front of option,
                    like here option is "nu" so for turning it off use "nonu".
:set nonu           removing line number display
:set hlsearch       This will highlight the matching word when we do search in VI editor,
                    quite useful but if you find it annoying or not able to see sometime
                    due to your color scheme you can turn it off by executing "set
                    nohlsearch".
:set wrap           If your file has contains some long lines and you want them to wrap
                    use this option, if its already on and you just don't want them to
                    wrap use "set nowrap".
:colorscheme        color scheme is used to change color of VIM editor, my favorite color
                    scheme is murphy so if you want to change color scheme of VI editor
                    you can do by executing "colorscheme murphy".
:syntax on          syntax can be turn on and off based on your need, if it's on it will
                    display color syntax for .xml, .html and .perl files.
:set ignorecase     This VI editor option allows you do case insensitive search because
                    if it's set VI will not distinguish between two words which are just
                    differ in case.
:set smartcase      Another VI editor option which allows case-sensitive search if the
                    word you are searching contains an uppercase character.
```

4.2.3 Navigation

Here are some navigating commands

gg	goes to start of file
SHIFT g	goes to end of file
0	goes to beginning of the line
$	goes to end of the line
nG	goes to nth line
:n	another way of going to nth line

4.2.4 Editing

Editing commands

yy	equivalent to cut also called yank
p	paste below line
SHIFT p	paste above line
dd	deletes the current line
5dd	deletes 5 lines
u	undo last change
CTRL + R	Re do last change

Copy (or cut) / paste (without strange indent)

1. move the mouse pointer to the beginning of your desired copy text

2. type 'v' (visual) for Visual mode, then using mouse pointer move to the end of selected text
3. type 'y' (yank) for Copy or 'd' (delete) for Cut
4. move to your paste location, then type 'p' (paste)

Tabulation

1. define TAB as 2 spaces

```
:set tabstop=2 shiftwidth=2 expandtab
```

2. replace TAB by 4 spaces

```
:%s/\t/    /g
```

4.2.5 Multi-files, multi-windows

Opening multi-files / another file

```
$ vim file1 file2 file3 ...
```

```
:n        edit next file among multi-files (with respect to the order given in the
          command line)
:wn       save the modification and edit the next file
:n!       edit the next file without saving the ongoing modification
:e        reload the current file
:e file   load file in the current window
```

Multi-windows

```
:sp(lit) file   split horizontally the window and load file in the splitted window
:vsplit file    split vertically the window and load file in the splitted window
:vs             vertically split window
CTRL + w + w    switch among all (sub-)windows
:q              close the current (sub-)window
```

4.2.6 Search and Replace

Searching commands

```
/Exception   will search for word "Exception" from top to bottom and stop when it got
             first match, to go to next match type "n" and for coming back to previous
             match press "Shift + N"
?Exception   will search for word "Exception" from bottom to top and stop when it got
             first match, to go to next match type "n" and for coming back to previous
             match press "Shift + N", remember for next match it will go towards top
             of file.
```

Find and replace

```
:%s/Old/New/g     This is an example of global search it will replace all occurrence of
                  word "Old" in file by "New". It's also equivalent to following
                  command ": 0,$ s/Old/New/g" which actually tells that search from
                  first to last line.
:%s/Old/New/gc    This is similar to first command but with the introduction of "c"; it
                  will ask for confirmation
:%s/Old/New/gci   This is command is global, case insensitive and ask for confirmation;
                  to make it case Sensitive use "I"
```

Substitution

Substitution is very useful when working with text. Below you have some example. For more information, you could check the link : http://vim.wikia.com/wiki/Search_and_replace

```
:s/abc/def/           change the first 'abc' of the line to 'def'
:s/abc/def/g          change all 'abc' of the line to 'def'
:%s/abc/def/g         change all 'abc' of all lines to 'def'
:%s/\<abc\>/def/g     change all words 'abc' of all lines to 'def'
:%s/\<abc\>/def/gI    change all words 'abc' (case sensitive) of all lines to 'def'
:%s/\<abc\>/def/gci   change all words 'abc' (case insensitive) of all lines to 'def',
                      ask for confirmation
:5,10s/abc/def/g      change all 'abc' to 'def', from line 5 to line 10 inclusive
:.,+5s/abc/def/g      change all 'abc' to 'def', for the current line and the 5 next
                      lines
:.,$s/abc/def/g       change all 'abc' to 'def', from the current line to the last line
:g/^a/s/abc/def/g     change all 'abc' to 'def', for each line starting with 'a'
```

4.3 JOE (Joe's Own Editor)

4.3.1 Basic commands

Launch JOE from command line

```
$ joe file.txt                    open and edit file.txt
$ joe -wordwrap file.txt          option wordwrap
$ joe -lmargin 5 -tab 5 file.txt  left margin = 5 chars and TAB = 5 chars
$ joe +25 file.txt                edit from 25th line
$ jmacs file.txt                  variant : simulate GNU-EMACS
$ jstar file.txt                  variant : simulate WordStar
$ jpico file.txt                  variant : simulate the Pine mailer editor PICO
$ rjoe file.txt                   variant : restraint the edit to the file file.txt
                                  only
```

Saving and quiting commands

CTRL + k + d	save the file
CTRL + k + x	save and exit
CTRL + c	exit without save
CTRL + k + z	exit and leave JOE in background (fg to go back)

Orthographe

CTRL + [+ n	check one word
CTRL + [+ l	check one file

Misc

CTRL + k + a	move to the middle
CTRL + t	display and choose the options
CTRL + r	refresh the display
CTRL + k + h	display or close the online help

4.3.2 Navigation

Cursor / Move

CTRL + b	move to left
CTRL + p	move to top
CTRL + f	move to right
CTRL + n	move to down
CTRL + z	move to the previous word
CTRL + x	move to the next word

Navigation

CTRL + u	previous screen
CTRL + v	next screen
CTRL + a	beginning of the line
CTRL + e	end of the line
CTRL + k + u	beginning of the file
CTRL + k + v	end of the file
CTRL + k + l	go to line n

4.3.3 Editing

Blocs operations

CTRL + k + b	beginning of the bloc
CTRL + k + k	end of the bloc
CTRL + k + m	move of the bloc
CTRL + k + c	copy the bloc
CTRL + k + w	write the bloc in a file
CTRL + k + y	delete the bloc
CTRL + k + /	filter the bloc

Deletion

CTRL + d	delete one character
CTRL + y	delete one line
CTRL + w	delete one word on the right of the cursor
CTRL + o	delete one word on the left of the cursor
CTRL + j	delete the rest of the line (i.e. the right side of the cursor)
CTRL + _	cancel the operation
CTRL + 6	redo the cancelled operation

Files

CTRL + k + e	open / edit a new file
CTRL + k + r	insert one file at the cursor position

4.3.4 Search

CTRL + k + f	search one text
CTRL + l	search the next

4.4 NANO (Nano's ANOther editor)

4.4.1 Basic commands

NANO is the open source clone of the editor PICO, distributed as part of the mail client Pine.

Launch NANO from command line

```
$ nano file.txt       open and edit file.txt
$ nano -B file.txt    save the original file as file.txt~ or ~.file
$ nano -m file.txt    activate the mouse cursor (if supported)
$ nano +25 file.txt   edit from the 25th line
$ jpico file.txt      simulator JOE of PICO
```

Short-cuts Fn

F1	CTRL + g	display online help (CTRL + x to quit)
F2	CTRL + x	quit NANO (or close ongoing buffer)
F3	CTRL + o	save ongoing file
F4	CTRL + j	reformat the text of paragraph
F5	CTRL + r	insert one file
F6	CTRL + w	search one text
F7	CTRL + y	previous screen
F8	CTRL + v	next screen
F9	CTRL + k	cut (and copy) the line (or the chosen text)
F10	CTRL + u	paste the cut text
F11	CTRL + c	display the cursor position
F12	CTRL + t	start the orthograph verification

Misc

CTRL + 6	choose one text from the cursor (CTRL + 6 to cancel the action)

4.4.2 Navigation

CTRL + _	go to line n and column m
CTRL + f	move to left
CTRL + b	move to right
CTRL + SPACE	move to the previous word
CTRL + p	previous line
CTRL + n	next line
CTRL + a	beginning of the line
CTRL + e	end of the line

`META` does not exist on most of the recent keyboards. We list the commands below just for reference.

Sometimes you could mimic a command `META` + `s` (toggle smooth scrolling mode on and off) as follows:

- press `Esc` key
- release `Esc` key
- press `s` key

META + SPACE	move to the next word
META + (	beginning of the paragraph
META +)	end of the paragraph
META + n	beginning of the file
META + /	end of the file
META +]	move to the opening curly bracket { which corresponds to the closing curly bracket }
META + =	move screen one line down
META + _	move screen one line up

4.4.3 Search

CTRL + n	search and replace one text

CHAPTER

FIVE

SYSTEM

5.1 Archive and Compression

5.1.1 tar (Tape ARchiver)

tar is very useful to backup / group your files. You could group your files / folders with

```
$ tar cvf backup.tar file1 file2 ... folder1 folder2 ...
```

To extract content of a tar file, just do the following

```
$ tar xvf backup.tar
```

N.B. tar is probably one of the first commands existing in the Unix / Linux world. You could invoke its options with a dash or not, this will not change the result.

tar options

Here is a list of some useful options :

c	create, for creating tar file
v	verbose, display name of files including,excluding from tar command
f	following, used to point name of tar file to be created.
	it actually tells tar command that name of the file is “next” letter just after options.
x	extract, for extracting files from tar file.
t	for viewing content of tar file
z	zip, tells tar command that create tar file using gzip.
j	another compressing option tells tar command to use bzip2 for compression
r	update or add file or directory in already existed .tar file
wildcards	to specify patters in unix tar command

tarball

When combining with gzip, tar becomes an extremely recommanded tool to archieve your documents. The resulted file is often called tarball, and is ended with either the extension .tgz or .tar.gz. For example, to archieve files / folders into a tarball:

```
$ tar zcvf backup.tgz file1 file2 ... folder1 folder2 ...
```

and to extract a tarball:

```
$ tar zxvf backup.tgz
```

creation

Create tar archive/file

```
$ tar -cvf tarball.tar *
$ tar -cvf tarball.tar file1 folder1 file2
```

Create compressed tar file (gzip, bzip2)

```
$ tar -zcvf tarball.tgz *
$ tar -zcvf tarball.tar.gz *
$ tar -jcvf tarball.tar.bz2 *
```

extraction

Extract contents from tar file

```
$ tar -xvf tarball.tar
```

Extract contents from compressed tar file

```
$ tar -zxvf tarball.tgz
$ tar -zxvf tarball.tar.gz
$ tar -jxvf tarball.tar.bz2
```

Extract a particular file from tar file

```
$ tar -xvf tarball.tar file
$ tar -zxvf tarball.tgz file
$ tar -zxvf tarball.tar.gz file
$ tar -jxvf tarball.tar.bz2 file
```

Extract a group of files from tar file

```
$ tar -xvf tarball.tar --wildcards "s*"
$ tar -zxvf tarball.tgz --wildcards "s*"
$ tar -zxvf tarball.tar.gz --wildcards "s*"
$ tar -jxvf tarball.tar.bz2 --wildcards "s*"
```

other operations

View contents of tar file

```
$ tar -tvf tarball.tar
```

View contents of compressed tar file

```
$ tar -ztvf tarball.tgz
$ tar -ztvf tarball.tar.gz
```

Update existing tar file (!not compressed tar file)

```
$ tar -cvf tarball.tar file folder1
$ tar -rvf tarball.tar folder2
```

Calculate the size (in KB) of (compressed) tar file

```
$ tar -cf - * | wc -c
$ tar -zcf - * | wc -c
$ tar -jcf - * | wc -c
```

Delete items (files/folders) from tar file (!not compressed)

```
$ gunzip tarball.tar.gz
$ tar --list --file tarball.tar
$ tar --file tarball.tar --delete file1 folder1 folder2/file2
$ tar --list --file tarball.tar
$ gzip tarball.tar
```

5.1.2 parallel gzip

For modern multi-processor, multi-core machines, a parallel implementation of gzip exists, called pigz. The official link is http://zlib.net/pigz/ and the manual could be downloaded here : http://zlib.net/pigz/pigz.pdf

To compress while keeping the original file, using

```
pigz --best -k file
```

and to decompress the .gz file

```
pigz -d file.gz
```

5.1.3 bz2

Compression to bz2

```
$ bzip2 file
$ bzip2 -v file
```

Decompression from bz2

```
$ bunzip2 file.bz2
$ bzip2 -d file.bz2
$ bunzip2 -v file.bz2
```

Archive to tarball .tar.bz2

```
$ tar cjvf files.tar.bz2 *.txt
$ tar -cjvf archive.tar.bz2 path/to/folder
$ tar --bzip2 -xf path/to/file.tar.bz2
$ tar -cvf - path/to/folder | bzip2 > archive.tar.bz2
```

Restore from tarball .tar.bz2

```
$ tar xjvf files.tar.bz2
$ tar -xjvf path/to/file.tar.bz2
$ bzcat path/to/file.tar.bz2 | tar -xvf -
```

5.1.4 zip

```
$ zip archive.zip file1 file2 file3
$ unzip archive.zip
```

5.1.5 lzma

Restore from lzma tarball

```
$ tar -xYvf archive.tar.lzma
```

5.1.6 lzo

The difference between the format .lzo and others (.gz or .bz2) are :

- .lzo is not installed by default on your machine;
- .lzo keeps the original file, unless if you use the option -U;
- .lzo runs fast, but the compression ratio is relatively low.

When we try to pass a list of files and folders to lzop, only the files will be compressed and the folders will be skipped.

Compression to lzo

```
$ lzop -v file
$ cat file | lzop > file.lzo
```

Delete the original file

```
$ lzop -U file
```

Test the result's integrality

```
$ lzop -t file.lzo
```

Show file headers

```
$ lzop --info file.lzo       afficher les en-têtes du ficher
```

Show compression information

```
$ lzop -l file.lzo
```

Show content of a compressed lzo file

```
$ lzop --ls file.lzo
```

Decompression from lzo

The lzo file is kept by default.

```
$ lzop -dv file.lzo
```

Archive all text files to a lzo tarball

```
$ tar --use-compress-program=lzop -cf files.tar.lzo *.txt
```

Restore from lzo tarball

```
$ tar --use-compress-program=lzop -xf files.tar.lzo
```

5.2 Daily Tools

5.2.1 find

```
find -print is the same as find, -print option is a default option
find -print0 ... | xargs -0 ... can avoid whitespace problem
find -delete can be used for -exec rm {} \;
```

Find out shell scripts without execution right

```
$ find . -iname "*.sh" -perm 644
```

Add execution right to shell scripts

```
$ for i in `find . -iname "*.sh" -perm 644`; do echo $i; chmod a+x $i; done;
```

Find out all files modified since less than 1 day, excatly 1 day or more than 1 day

```
$ find . -mtime -1
$ find . -mtime 1
$ find . -mtime +1
```

Delete found files

```
$ find . -name "*.tmp" -delete
$ find . -name "*.tmp" -print | xargs rm -f
```

Grep found files using -print0 and xargs -0 to avoid whitespace problem

```
$ find . -name "*.txt" -print | xargs grep "Exception"
$ find . -name "*.txt" -print0 | xargs -0 grep "Exception"
```

Find in the current folder, type file (not link, directory) and newer than first_file

```
$ find . -maxdepth 1 -type f -newer first_file
```

Find in the current folder, type file and modified since more than 15 mins

```
$ find . -type f -cmin 15 -prune
```

Find out and list files more than 1000 bytes

```
$ find . -size +1000c -exec ls -l {} \;
```

Find out files with size more than 10000 bytes and less than 50000 bytes

```
$ find . -size +10000c -size -50000c -print
```

Find out and list files modified 10 days ago and more than 50000 bytes

```
$ find . -mtime +10 -size +50000c -exec ls -l {} \;
```

Find out and list all symbolic links

```
$ find . -type l -print | xargs ls -ld | awk '{print $9 " " $10 " " $11}'
```

Find all the files without permission 777

```
$ find / -type f ! -perm 777
```

Find all the SGID bit files whose permissions set to 644

```
$ find / -perm 2644
```

Find all the Sticky Bit set files whose permission are 551

```
$ find / -perm 1551
```

Find all SUID set files.

```
$ find / -perm /u=s
```

Find all SGID set files.

```
$ find / -perm /g+s
```

Find all empty files under certain path

```
$ find /tmp -type f -empty
```

Find all all empty directories under certain path

```
$ find /tmp -type d -empty
```

5.2.2 grep

There are several variants of grep:

- grep
- egrep : extended grep
- fgrep : fixed grep
- zgrep

Grep A, but excluding B

```
$ grep A file | grep -v B
```

Count the occurence of a word A

```
$ grep -c A file
```

Grep A and show n lines prior to the A position and n lines after the A position

```
$ grep --context=n A file
$ grep -C n A file
```

Extended grep with more regular expression

```
$ egrep "A|B" file
```

Case insensitive grep

```
$ grep -i A file
```

Grep .gz file

```
$ zgrep A file.gz
```

Grep whole word

```
$ grep -w WORD file
```

Grep words starting with WORD

```
$ grep '<WORD' file
```

Grep words ending in WORD

```
$ grep 'WORD>' file
```

Grep with matching pattern in color

```
$ grep A file --color
```

Grep among files matching the given pattern

```
$ grep -l A *.log
```

Grep showing line number

```
$ grep -n A file
```

Recursive grep

```
$ grep -R A folder
```

5.2.3 sort

Sort based on the numerical order of the 2nd column

```
$ ps -ef | sort -nk2
```

Sort based on the reverse numerical order of the 3rd column

```
$ ps -ef | sort -rnk3
```

Sort based on the alphabetic order of the 4th column

```
$ ps -ef | sort -nk4
```

Sort based on the alphabetic order

```
$ cat file | sort
$ sort file
```

Sort based on the alphabetic order, and remove duplicates

```
$ cat file | sort | uniq
$ cat file | sort -u
$ sort -u file
```

Sort file, case insensitive

```
$ sort -f file
```

5.2.4 sed

Replace word A by B in a file

```
$ sed s/A/B/g file
```

5.2.5 cat

Create a new file and fill it

```
$ cat > file.txt
fill the file
...
CTRL+D
```

Append new input to a file (so, keep its current content)

```
$ cat >> file.txt
append the file
...
CTRL+D
```

Displaying with line numbers (option -b will number only non-empty lines)

```
$ cat -n file.txt
$ cat -b file.txt
```

Copy files

```
$ cat file1 > file2
```

Concatenating files

```
$ cat file 1 file2 > file
```

Squeezing repeating blank lines into one

```
$ cat -s file
```

Show non-printing chars (^V, ^M, etc.)

```
$ cat -v file
```

5.2.6 top / htop

By default, the display of top is ordered by CPU usage

```
$ top
```

In the top interface, you have the following choices

M	ordered by Memory usage
1	display all CPUs
c	display path of commands
k	kill a process via its PID
z	highlight running processes
s	change refresh delay
h or ?	online help
q	quit
W	save changes (into ~/.toprc)

To work on one particular user

```
$ top -u user
```

5.2.7 lsof

list all open files

```
$ lsof
```

`FD` stands for **File descriptor** and may seen some of the values as:

cwd	current working directory
rtd	root directory
txt	program text (code and data)
mem	memory-mapped file

Also in `FD` column numbers like `1u` is actual file descriptor and followed by u,r,w of it's mode as:

r	for read access
w	for write access
u	for read and write access

`TYPE` of files and it's identification:

DIR	Directory
REG	Regular file
CHR	Character special file
FIFO	First In First Out

list open files of one user

```
$ lsof -u user
```

find out all the running process of specific port

```
$ lsof -i TCP:22
```

shows only IPv4 and IPv6 network files

```
$ lsof -i 4
$ lsof -i 6
```

list all running process of open files of TCP Port ranges from 1-1024.

```
$ lsof -i TCP:1-1024
```

exclude user with '^' Character : e.g. to exclude root user

```
$ lsof -i -u^root
```

find out a specific user is looking what files and commands

```
$ lsof -i -u user
```

list all network connections : option '-i' shows the list of all network connections 'LISTENING & ESTABLISHED'.

```
$ lsof -i
```

search by PID

```
$ lsof -p 1
```

kill all activity of particular user

```
$ kill -9 `lsof -t -u user`
```

5.2.8 netstat

listing all the LISTENING Ports of TCP and UDP connections

```
$ netstat -a | more
```

listing only TCP (Transmission Control Protocol) port connections

```
$ netstat -at
```

listing only UDP (User Datagram Protocol) port connections

```
$ netstat -au
```

listing all active listening ports connections

```
$ netstat -l
```

listing all active listening TCP ports

```
$ netstat -lt
```

listing all active listening UDP ports

```
$ netstat -lu
```

Listing all active UNIX listening ports

```
$ netstat -lx
```

Showing Statistics by Protocol

```
$ netstat -s
```

Showing statistics of TCP protocol

```
$ netstat -st
```

Showing statistics of UDP Protocol

```
$ netstat -su
```

Displaying service name with their PID number : option -tp will display "PID/Program Name"

```
$ netstat -tp
```

Displaying Promiscuous mode with -ac switch, netstat print the selected information or refresh screen every five second. Default screen refresh in every second.

```
$ netstat -ac 5 | grep tcp
```

Displaying Kernel IP routing table

```
$ netstat -r
```

Showing network interface packet transactions including both transferring and receiving packets with MTU size

```
$ netstat -i
```

Showing Kernel interface table, similar to ifconfig command.

```
$ netstat -ie
```

Displaying multicast group membership information for both IPv4 and IPv6

```
$ netstat -g
```

Print netstat information every few second

```
$ netstat -c
```

Finding un-configured address families with some useful information

```
$ netstat --verbose
```

Find out how many listening programs running on a port

```
$ netstat -ap | grep http
```

Displaying RAW Network Statistics

```
$ netstat --statistics --raw
```

5.2.9 screen

To use screen

```
$ screen
```

When connection lost or crash, to restore previous sessions (very useful !)

```
$ screen -r
```

Remove dead screen

```
$ screen -wipe
```

Short-cuts

CTRL + a then CTRL + c	CTRL + a, c	Create new window
CTRL + a then 0		Select Window 0
CTRL + a then 1		Select Window 1
CTRL + a then 2		Select Window 2
CTRL + a then 3		Select Window 3
CTRL + a then 4		Select Window 4
CTRL + a then 5		Select Window 5
CTRL + a then 6		Select Window 6
CTRL + a then 7		Select Window 7
CTRL + a then 8		Select Window 8
CTRL + a then 9		Select Window 9
CTRL + a then A		Set the name of the current window
CTRL + D		Close the current Window

config file .screenrc

```
# term vt100
# termcap sun-color bc=
# terminfo sun-color bc=

defscrollback 3000
startup_message off
bind '^\'

termcapinfo xterm ti@:te@

termcap  vt100 'AF=\E[3%dm:AB=\E[4%dm'
terminfo vt100 'AF=\E[3%p1%dm:AB=\E[4%p1%dm'
termcap  xterm 'AF=\E[3%dm:AB=\E[4%dm'
terminfo xterm 'AF=\E[3%p1%dm:AB=\E[4%p1%dm'

# test
hardstatus alwayslastline "%{bw}x: %-Lw%{= bw}%50>%n%f* %t%{-}%+Lw%< %>"

# Bind F11 and F12 (NOT F1 and F2) to previous and next screen window
bindkey -k F1 prev
bindkey -k F2 next

# Produce a bell sound even when a background window is belling
bell_msg 'Bell in %^G'
```

5.2.10 MD5

`md5sum` stands for **Compute and Check MD5 Message Digest**

md5 checksum (commonly called hash) is used to match or verify integrity of files that may have changed as a result of a faulty file transfer, a disk error or non-malicious interference.

```
$ md5sum file
```

5.2.11 dd

`dd` stands for **Convert and Copy a file**

Can be used to convert and copy a file and most of the times is used to copy a iso file (or any other file) to a usb device (or any other location), thus can be used to make a 'Bootlable' Usb Stick.

```
$ dd if=/home/user/Downloads/debian.iso of=/dev/sdb1 bs=512M; sync
```

5.2.12 Calendar

```
$ cal 02 1835
   February 1835
Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
```

5.2.13 Date

```
$ date
Fri May 17 14:13:29 IST 2013
```

5.3 Windows

5.3.1 Connect to Internet via Ethernet cable (from PC/laptop)

Control Panel –> Network and Internet –> Network Connections

Ctrl + select local and wireless connections, right click **Bridge Connections**

CHAPTER

SIX

MATHMATICS

6.1 Algebra

6.2 Geometry