
technotes Documentation

Release 1

Tom JIANG

April 09, 2015

CONTENTS

1	Kid's activity	1
1.1	Minecraft Pi Edition	1
1.2	Pygame	7
1.3	Scratch	8
2	Hardware	9
2.1	Raspberry Pi	9
2.2	Arduino	11
3	System	13
3.1	Linux	13
3.2	Windows	13
4	Editor	15
4.1	VIM (Vi IMproved)	15
4.2	JOE (Joe's Own Editor)	18
4.3	NANO (Nano's ANOther editor)	20
5	Programming language	23
5.1	Shell	23
5.2	Python	23
6	Mathmatics	25
6.1	Algebra	25
6.2	Geometry	25

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

```

1 from mcpi import minecraft
2
3 mc = minecraft.Minecraft.create()
4
5 x,y,z = mc.player.getTilePos()
6 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.

```

1 from mcpi import minecraft
2
3 mc = minecraft.Minecraft.create()
4
5 x,y,z = mc.player.getTilePos()
6 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.

```

1 from mcpi import minecraft
2
3 mc = minecraft.Minecraft.create()
4
5 x,y,z = mc.player.getTilePos()
6
7 tnt = 46
8 activated = 1
9 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.

```
1 from mcpi import minecraft
2 from time import sleep
3
4 mc = minecraft.Minecraft.create()
5
6 grass = 2
7 flower = 38
8 while True:
9     x,y,z = mc.player.getTilePos()
10    block_beneath = mc.getBlock(x,y-1,z)
11    if block_beneath == grass:
12        mc.setBlock(x,y,z,flower)
13    else:
14        mc.setBlock(x,y-1,z,grass)
15    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.

```
1 from mcpi import minecraft, block
2
3 mc = minecraft.Minecraft.create()
4
5 x,y,z = mc.player.getTilePos()
6 size = int(raw_input("size of area to clear? "))
7 if size > 0:
8     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

```
1 from mcpi import minecraft, block
2
3 mc = minecraft.Minecraft.create()
4 SIZE = 20
5
6 def house():
7     midx = x + SIZE/2
8     midy = y + SIZE/2
9     mc.setBlocks(x, y, z, x+SIZE, y+SIZE, z+SIZE, block.COBBLESTONE.id)
10    mc.setBlocks(x+1, y+1, z+1, x+SIZE-1, y+SIZE-1, z+SIZE-1, block.AIR.id)
11    mc.setBlocks(x+1, y+1, z+1, x+SIZE-1, y+1, z+SIZE-1, block.WOOL.id, 7)
12    # left window
13    mc.setBlocks(x+3, y+SIZE-3, z, midx-3, midy+3, z, block.GLASS.id)
14    # right window
15    mc.setBlocks(midx+3, y+SIZE-3, z, x+SIZE-3, midy+3, z, block.GLASS.id)
16    # door
17    mc.setBlocks(midx-3, y, z, midx+3, midy, z, block.AIR.id)
```



```

18     # roof
19     mc.setBlocks(      x,y+SIZE+1,  z,  x+SIZE,y+SIZE+1,  z+SIZE,      block.SNOW.id)
20
21 x,y,z = mc.player.getTilePos()
22
23 # build a house
24 house()

```

Build a street

```

1  from mcpi import minecraft, block
2
3  mc = minecraft.Minecraft.create()
4  SIZE = 20
5
6  def house():
7      midx = x + SIZE/2
8      midy = y + SIZE/2
9      mc.setBlocks(      x,          y,  z,  x+SIZE,  y+SIZE,  z+SIZE,block.COBBLESTONE.id)
10     mc.setBlocks(      x+1,        y+1,z+1,x+SIZE-1,y+SIZE-1,z+SIZE-1,      block.AIR.id)
11     mc.setBlocks(      x+1,        y+1,z+1,x+SIZE-1,      y+1,z+SIZE-1,      block.WOOL.id,7)
12     # left window
13     mc.setBlocks(      x+3,y+SIZE-3,  z,  midx-3,  midy+3,      z,      block.GLASS.id)
14     # right window
15     mc.setBlocks(midx+3,y+SIZE-3,  z,x+SIZE-3,  midy+3,      z,      block.GLASS.id)
16     # door
17     mc.setBlocks(midx-3,      y,  z,  midx+3,  midy,      z,      block.AIR.id)
18     # roof
19     mc.setBlocks(      x,y+SIZE+1,  z,  x+SIZE,y+SIZE+1,  z+SIZE,      block.SNOW.id)
20
21 x,y,z = mc.player.getTilePos()
22
23 # build a street
24 for h in range(5):
25     house()
26     x += SIZE

```

`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.

```

1  from mcpi import minecraft, block
2  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()

```

Treasure Hunt

```

1  from mcpi import minecraft, block
2  import time, random
3
4  mc = minecraft.Minecraft.create()
5
6  score = 0
7  range = 5 # increase this number to make a more difficult game!
8
9  treasurex = None
10
11  def placeTreasure():
12      pass

```

```

13
14 def checkHit():
15     pass
16
17 def homingBeacon():
18     pass
19
20 bridge = []
21
22 def buildBridge():
23     pass
24
25 while True:
26     time.sleep(0.01)
27
28     if treasurex == None and len(bridge) == 0:
29         placeTreasure()
30
31     checkHit()
32     homingBeacon()
33     buildBridge()

```

1.2 Pygame

1.2.1 List of pygame programs

Draw a circle

```

1 import pygame
2
3 width,height = 640,480
4 radius = 100
5 fill = 1
6
7 pygame.init()
8 window = pygame.display.set_mode((width,height))
9 window.fill(pygame.Color(255,255,255)) # white
10
11 while True:
12     pygame.draw.circle(window,
13                         pygame.Color(255,0,0), # red
14                         (width/2,height/2),
15                         radius,
16                         fill)
17     pygame.display.update()
18     if pygame.QUIT in [e.type for e in pygame.event.get()]:
19         break

```

Draw circles based on mouse move / position

```

1 import pygame
2 from pygame.locals import *
3

```

```
4 width,height = 640,640
5 radius = 0
6 fill = 1
7 mouseX,mouseY = 0,0
8
9 pygame.init()
10 window = pygame.display.set_mode((width,height))
11 window.fill(pygame.Color(255,255,255)) # white
12 fps = pygame.time.Clock() # FPS = Frame Per Second
13
14 while True: # one frame per loop
15     for event in pygame.event.get():
16         if event.type == MOUSEMOTION:
17             mouseX,mouseY = event.pos
18         if event.type == MOUSEBUTTONDOWN: # mouse click
19             window.fill(pygame.Color(255,255,255)) # clear screen
20             radius = (abs(width/2 - mouseX) + abs(height/2 - mouseY))/2 + 1
21             pygame.draw.circle(window,
22                               pygame.Color(255,0,0), # red
23                               (mouseX,mouseY),
24                               radius,
25                               fill)
26     pygame.display.update()
27     if pygame.QUIT in [e.type for e in pygame.event.get()]:
28         break
29     fps.tick(30) # wait so that frame rate is 30 fps
```

1.3 Scratch

HARDWARE

2.1 Raspberry Pi

2.1.1 Default settings

login	pi
password	raspberrypi
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 --subsystem-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

3.1 Linux

3.2 Windows

3.2.1 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**

4.1 VIM (Vi IMproved)

4.1.1 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 want 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 quitting 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**

4.1.2 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 .
:col-orscheme	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 ig-norecase	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 smart-case	Another VI editor option which allows case-sensitive search if the word you are searching contains an uppercase character.

4.1.3 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

4.1.4 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
```

4.1.5 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
```

4.1.6 Search and Replace

Searching commands

<code>/Exception</code>	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"
<code>?Exception</code>	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

<code>:%s/Old/New/g</code>	This is an example of global search it will replace all occurrence of word "Old" in file with word "New". Its also equivalent to following command <code>: 0,\$ s/Old/New/g</code> which actually tells that search from first to last line.
<code>:%s/Old/New/gc</code>	This is similar to first command but with the introduction of "c" it will ask for confirmation
<code>:%s/Old/New/gci</code>	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

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

4.2 JOE (Joe's Own Editor)

4.2.1 Basic commands

Launch JOE from command line

<code>\$ joe file.txt</code>	open and edit file.txt
<code>\$ joe -wordwrap file.txt</code>	option wordwrap
<code>\$ joe -lmargin 5 -tab 5 file.txt</code>	left margin = 5 chars and TAB = 5 chars
<code>\$ joe +25 file.txt</code>	edit from 25th line
<code>\$ jmacs file.txt</code>	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 quitting 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

4.2.2 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

4.2.3 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

4.2.4 Search

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

4.3 NANO (Nano's ANOther editor)

4.3.1 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 cloase 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)
----------	---

PROGRAMMING LANGUAGE

5.1 Shell

5.2 Python

MATHEMATICS

6.1 Algebra

6.2 Geometry