

Bugnon / minecraft

Branch: master ▾ [minecraft / virtual_reality / notebook / Notebook_Virtual_Reality.ipynb](#)

[Find file](#) [Copy path](#)

 BenJonas translocated file

29e1f6f 5 hours ago

0 contributors

727 lines (726 sloc) | 35.1 KB

Entrez physiquement dans le monde de Minecraft!

Comme projet de l'année en informatique, nous avons décidé d'essayer de créer une **réalité virtuelle dans Minecraft** à l'aide d'un **Raspberry Pi**. Après de nombreuses heures de travail, par des moyens très simple, nous avons réussi à créer une façon d'**incarner réellement le personnage du monde de Minecraft** et de le faire bouger dans le jeu en bougeant dans la réalité. Notre projet est donc un prototype de réalité virtuelle créé à l'aide des connaissances que nous avons acquises durant une année d'option complémentaire d'informatique et les moyens mis à disposition par le gymmnase.

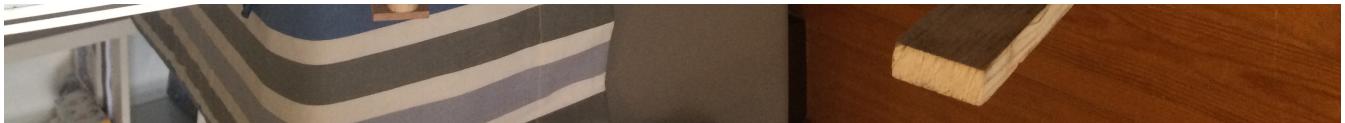
Nous avons soumis notre projet au **concours Bugnplay**, le plus grand concours d'informatique de Suisse, où nous avons été invités à présenter notre projet par la suite lors de la **finale du concours**. Le côté **original** et **amusant** de notre projet nous a permis d'y remporter deux prix, à savoir le **Best Classroom Project** et le **Senior Swiss Space Award**.

Description de l'installation

Nous avons créer une installation plutôt complexe avec un **Raspberry Pi**, un **écran**, **2 souris**, **1 batterie externe**(5.5 V), **2 plaque GPIO**, **6 boutons**, **3 mini-servos**, **une plaque en bois** à placer sous les pieds et surtout une **armure** que nous avons construit nous même.

Voici une image de notre exo-squelette vu de devant.





Et en voici une vu de derrière.



Déplacement

Pour ce qui est du déplacement du personnage, nous avons créé une plaque sur laquelle nous avons fixé deux boutons. Lorsque l'on marche sur cette plaque, le personnage marche dans le monde de Minecraft. Le principe est le suivant: lorsque l'un des deux boutons est enfoncé, la touche w destinée à faire avancer le joueur est simulée. Lorsque l'on appuie sur rien ou sur les deux boutons, rien ne se passe.

Voilà la plaquette que nous avons construit. Il y a deux emplacements pour les pieds du joueur.

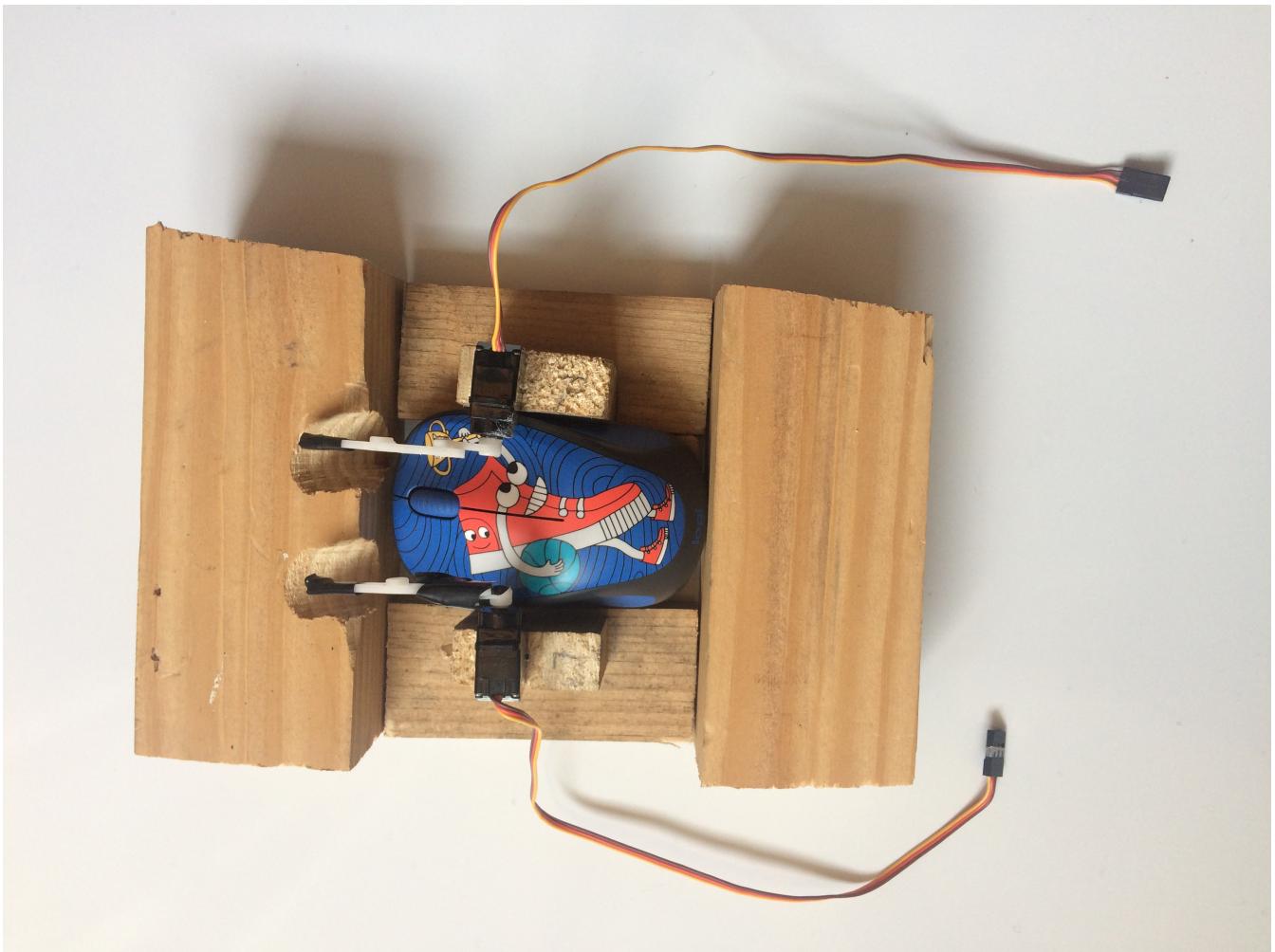




Construire et casser des briques

Afin de construire et casser des briques, nous avons posé sur notre armure 2 boutons au niveau des mains destinés à construire et à casser des briques. Chaque bouton est relié à un mini-servo qui sont placés de tel façon à cliquer respectivement sur le bouton droit et le bouton gauche de la souris lorsque ils sont sollicités. Lorsque l'on appuie sur ces boutons dans la réalité, les servos sont sollicités de tel sorte à cliquer sur la souris pour créer ou détruire des blocs dans le monde de Minecraft.

Sur cette image, il est possible de voir la construction que nous avons réalisé pour placer les 2 mini-servos.



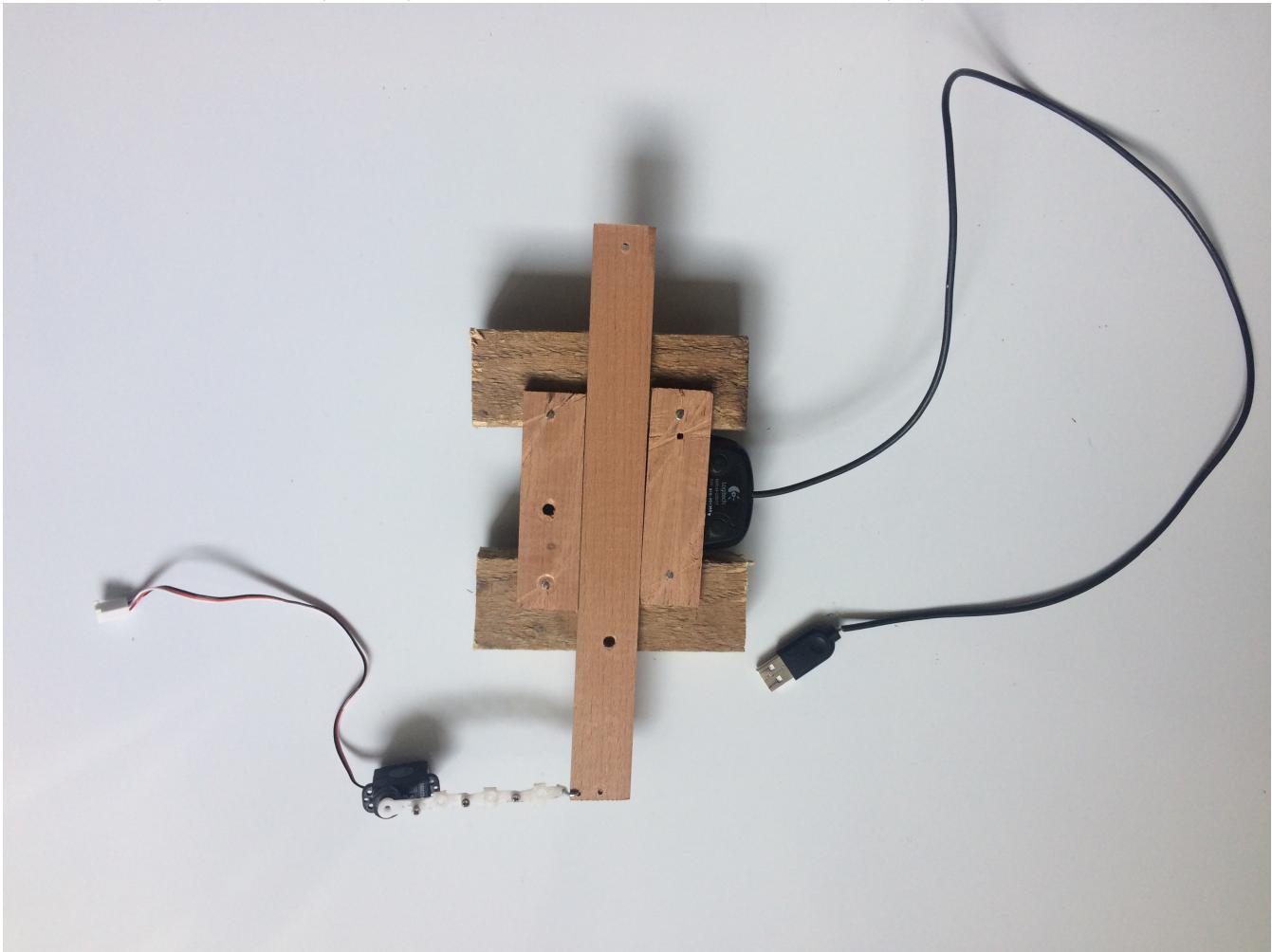
Nous avons du simuler le clique de la souris dans la réalité, car il n'était pas possible de le simuler virtuellement. Le module pyautogui permet

de simuler un clique de la souris mais cela ne fonctionnait pas dans Minecraft.

Rotation du personnage

Pour pouvoir tourner dans le monde de Minecraft en tournant dans la réalité, nous avons placé 2 boutons au niveau de la tête de telle façon à lorsque nous tournons la tête à droite (resp. à gauche), le bouton droit (resp. gauche) est appuyé. Ces boutons provoquent une rotation à droite (resp. à gauche) d'un troisième mini-servo. Celui-ci est attaché à une plaque qui bouge sous une souris. La souris détecte donc un mouvement vers la droite (resp. gauche) et fait tourner le personnage vers la droite (resp. gauche).

Voici l'installation pour la rotation du personnage. La souris est coincée entre deux blocs de bois et la plaque coulissante est attachée au servo.



Nous avons du simuler le mouvement de la souris dans la réalité, car la version de Minecraft que nous avions ne proposait pas de façon de simuler la rotation du personnage. De plus, la fonction du module `pyautogui` qui permettait de simuler virtuellement le mouvement de la souris ne fonctionnait pas dans Minecraft.

Code

Importation des modules nécessaires.

```
In [1]: import time
import pyautogui
import RPi.GPIO as gpio
from time import sleep
from mcpi.minecraft import Minecraft
```

```
-----
ImportError                                                 Traceback (most recent call last)
<ipython-input-1-4c997f1c2ff5> in <module>()
      1 import time
----> 2 import pyautogui
      3 import RPi.GPIO as gpio
```

```

4 from time import sleep
5 from mcpi.minecraft import Minecraft

ImportError: No module named 'pyautogui'

```

Définition des boutons sur le GPIO

```

In [2]: # Boutons pour avancer (plaques des pieds)
buttonL = 14
buttonR = 15

# Boutons pour créer et casser brique
buttonA = 23 # clique gauche souris, connecter à pwmL
buttonB = 24 # clique droit souris, connecter à pwmR

# Boutons pour la rotation du personnage, connectés à pwm
buttonC = 20
buttonD = 21

```

Position du joueur sur Minecraft

```

In [3]: mc = Minecraft.create()
x, y, z = mc.player.getPos()

-----
NameError                                                 Traceback (most recent call last)
<ipython-input-3-9c95f066e716> in <module>()
----> 1 mc = Minecraft.create()
      2 x, y, z = mc.player.getPos()

NameError: name 'Minecraft' is not defined

```

Définition des fonctions servant à créer le monde idéal pour notre projet

```

In [4]: def monde_plat():
    """Créer un monde plat pour faciliter déplacement."""
    mc.setBlocks(x-300, y, z-300, x+300, y, z+300, 2)
    mc.setBlocks(x-300, y+1, z-300, x+300, y+300, z+300, 0)

def arbre(a, b, c):
    """Créer un arbre basé sur la position actuelle du joueur."""
    mc.setBlocks(x+a, y, z+c, x+a, y+b, z+c, 17)
    mc.setBlocks(x+(a-2), y+b, z+(c-2), x+(a+2), y+b, z+(c+2), 18)
    mc.setBlocks(x+(a-1), y+(b+1), z+(c-1), x+(a+1), y+(b+1), z+(c+1), 18)
    mc.setBlocks(x+a, y+(b+2), z+c, x+a, y+(b+2), z+c, 18)

def foret():
    """Crée une forêt avec 12 arbres."""
    arbre(6, 4, 4)
    arbre(10, 3, 4)
    arbre(20, 4, 8)
    arbre(3, 8, 9)
    arbre(5, 9, 9)
    arbre(7, 7, 7)
    arbre(3, 9, 67)
    arbre(50, 10, 34)
    arbre(65, 3, 56)
    arbre(34, 13, 90)
    arbre(90, 7, 45)
    arbre(27, 8, 74)
    arbre(78, 4, 5)

```

Fonctions servant à initialiser servos et boutons

```

In [5]: def init_boutons():
    """Initialisation des 6 boutons en pull-up."""
    gpio.setmode(gpio.BCM) # Mode du GPIO
    gpio.setup(buttonL, gpio.IN, pull_up_down=gpio.PUD_UP)
    gpio.setup(buttonR, gpio.IN, pull_up_down=gpio.PUD_UP)
    gpio.setup(buttonA, gpio.IN, pull_up_down=gpio.PUD_UP)
    gpio.setup(buttonB, gpio.IN, pull_up_down=gpio.PUD_UP)
    #gpio.setup(buttonC, gpio.IN, pull_up_down=gpio.PUD_UP)
    #gpio.setup(buttonD, gpio.IN, pull_up_down=gpio.PUD_UP)

```

```

    gpio.setup(buttonC, gpio.IN, pull_up_down=gpio.PUD_UP)
    gpio.setup(buttonD, gpio.IN, pull_up_down=gpio.PUD_UP)

def init_servos():
    """Initialiser les servo-moteurs."""
    #Servo pour rotation du personnage, connecté à boutons C et D
    gpio.setup(2, gpio.OUT)
    pwm = gpio.PWM(2, 50)

    # Servo pour clique gauche, connecté à bouton A
    gpio.setup(18, gpio.OUT)
    pwmL = gpio.PWM(18, 50)

    # Servo pour clique droit, connecté à bouton B
    gpio.setup(13, gpio.OUT)
    pwmR = gpio.PWM(13, 50)

    ##Etat de base des servo
    pwmL.start(4)# Position de base du servo pwmL
    pwmR.start(8)# Etat de base du servo pwmR

    x = 7.5
    pwm.start(x)

```

Initialisation à l'état précédent des boutons.

```
In [6]: L0 = False
R0 = False
A0 = False
B0 = False
C0 = False
D0 = False
```

Fonctions qui initialise

```
In [8]: init_boutons()
init_servos()
monde_plat()
foret()

-----
NameError Traceback (most recent call last)
<ipython-input-8-67076539d178> in <module>()
----> 1 init_boutons()
      2 init_servos()
      3 monde_plat()
      4 foret()

<ipython-input-5-b32b54f5b77a> in init_boutons()
      1 def init_boutons():
      2     """Initialisation des 6 boutons en pull-up."""
----> 3     gpio.setmode(gpio.BCM) # Mode du GPIO
      4     gpio.setup(buttonL, gpio.IN, pull_up_down=gpio.PUD_UP)
      5     gpio.setup(buttonR, gpio.IN, pull_up_down=gpio.PUD_UP)

NameError: name 'gpio' is not defined
```

Boucle

```
In [1]: while True:
##  Code avancer buttons

    L = gpio.input(buttonL)
    R = gpio.input(buttonR)

    if not L and L0:
        pyautogui.keyDown('w')
    if L and not L0:
        pyautogui.keyUp('w')
    L0 = L

    if not R and R0:
        pyautogui.keyDown('w')
    if R and not R0:
```

```

    if A and not A0:
        pyautogui.keyUp('w')
    R0 = R # annulation de la boucle

## Code créer et casser brique avec servo pwmL et pwmR button A et button B

A = gpio.input(buttonA)
B = gpio.input(buttonB)

# clique gauche de la souris, casser une brique

if not A and A0:
    pwmL.ChangeDutyCycle(2.5)
    time.sleep(0.5)
    pwmL.ChangeDutyCycle(4)
if A and not A0:
    pass
A0 = A

# clique droite de la souris, créer une brique

if not B and B0:
    pwmR.ChangeDutyCycle(11)
    time.sleep(0.75)
    pwmR.ChangeDutyCycle(8)
if B and not B0:
    pass
B0 = B

## code rotation personnage pwm et buttonC et buttonD

C = gpio.input(buttonC)
D = gpio.input(buttonD)

### tire le miservo vers 12.5

if not C and C0 and x < 12.5:
    pwm.ChangeDutyCycle(x+2.5)
    x = x+2.5
if C and not C0:
    pass
C0 = C

# tire le mini servo vers 2.5

if not D and D0 and x > 2.5:
    pwm.ChangeDutyCycle(x-2.5)
    x = x-2.5
if D and not D0:
    pass
D0 = D

```

```

-----
NameError                                                 Traceback (most recent call last)
<ipython-input-1-abddd6b08ff3> in <module>()
      2 ##  Code avancer buttons
      3
----> 4     L = gpio.input(buttonL)
      5     R = gpio.input(buttonR)
      6
NameError: name 'gpio' is not defined

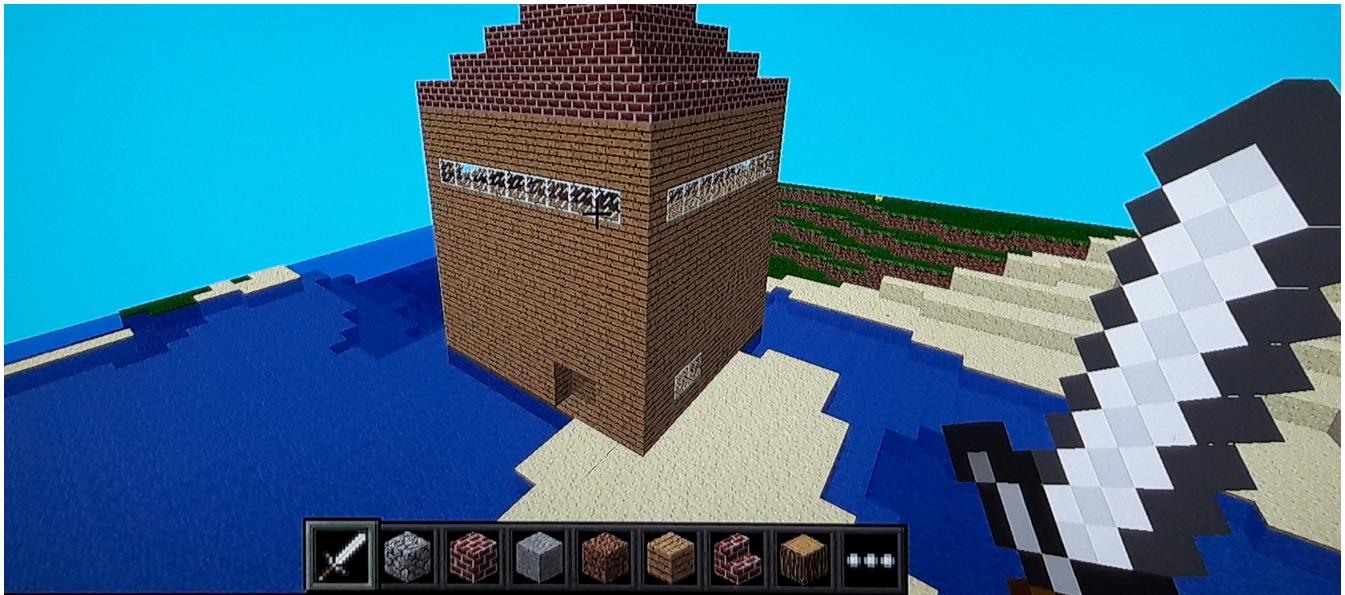
```

Structures que nous avons créé

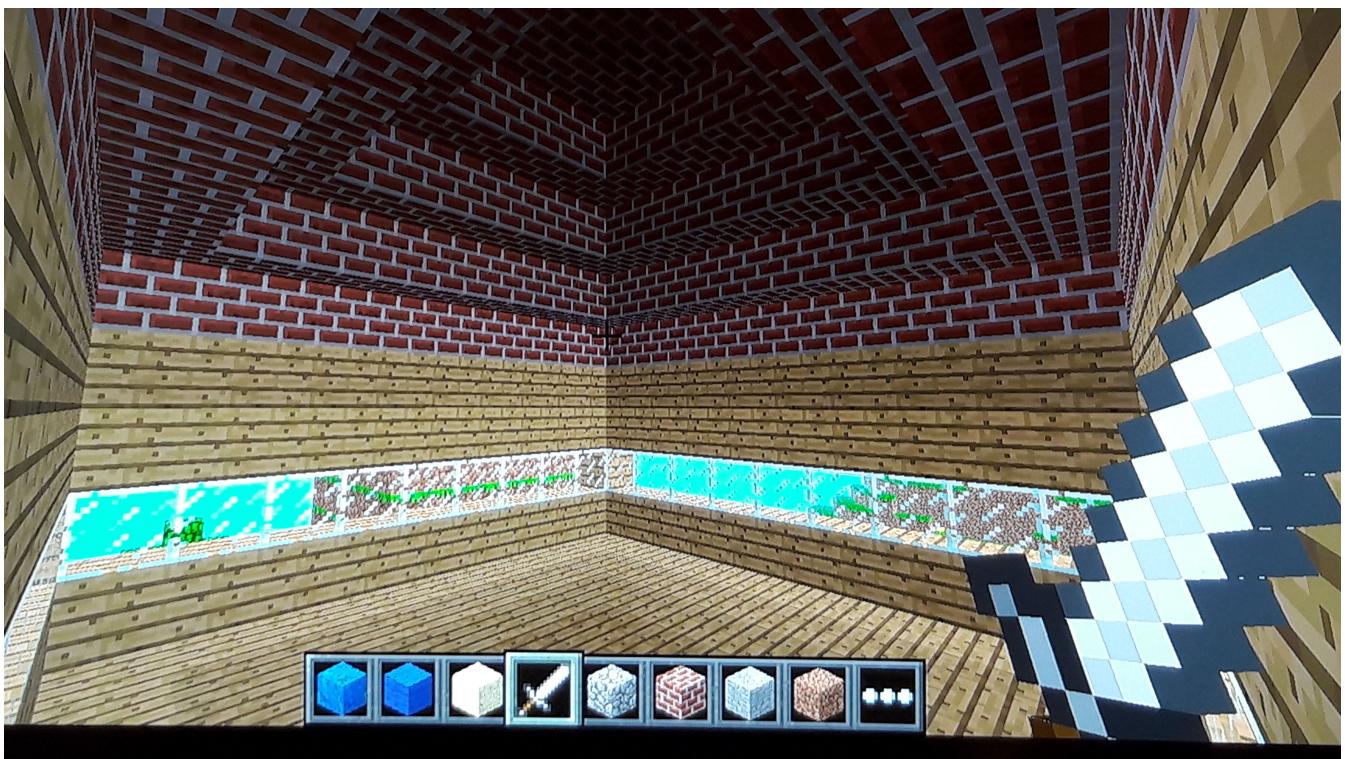
Nous avons créé une **maison** à 2 étages avec une verranda luxuriante offrant une vue panoramique sur le monde de Minecraft. Cette maison dispose de fenêtres, de torches et d'une échelle. Le toit d'architecture japonaise est fait en argile.

Voici une photo de l'extérieur de la maison:





Et voici l'intérieur:



Et voici le code:

```
In [3]: from mcpi.minecraft import Minecraft
from mcpi import block
mc = Minecraft.create()
x, y, z = mc.player.getPos()
planches = block.WOOD_PLANKS

def limites_maison():
    mc.setBlocks(x+2, y-1, z+2, x+12, y+10, z+12, planches)
    mc.setBlocks(x+3, y, z+3, x+11, y+9, z+11, 0)

def murs_intérieurs ():
    mc.setBlocks(x+2, y, z+7, x+12, y+5, z+7, planches)
    mc.setBlocks(x+7, y, z+7, x+7, y+5, z+12, planches)

def portes():
    mc.setBlocks(x+4,y,z+2,x+5,y+1,z+2, 0)
    mc.setBlocks(x+4, y, z+7, x+4, y+1, z+7, 0)
    mc.setBlocks(x+9, y, z+7, x+9, y+1, z+7, 0)
```

```

def passage_niveau():
    mc.setBlocks(x+2, y+6, z+2, x+12, y+6, z+12, planches)
    mc.setBlocks(x+3, y+6, z+9, x+3, y+6, z+11, 0)

def vitres():
    mc.setBlocks(x+2, y+1, z+4, x+2, y+1, z+5, 20)
    mc.setBlocks(x+12, y+1, z+4, x+12, y+1, z+5, 20)
    mc.setBlocks(x+3, y+8, z+2, x+11, y+8, z+2, 20)
    mc.setBlocks(x+2, y+8, z+3, x+2, y+8, z+11, 20)
    mc.setBlocks(x+12, y+8, z+3, x+12, y+8, z+11, 20)
    mc.setBlocks(x+3, y+8, z+12, x+11, y+8, z+12, 20)

def toit():
    mc.setBlocks(x+3, y+10, z+3, x+11, y+10, z+11, 0)
    mc.setBlocks(x+2, y+11, z+2, x+12, y+11, z+12, 45)
    mc.setBlocks(x+3, y+11, z+3, x+11, y+11, z+11, 0)
    mc.setBlocks(x+3, y+12, z+3, x+12, y+12, z+11, 45)
    mc.setBlocks(x+4, y+12, z+4, x+10, y+12, z+10, 0)
    mc.setBlocks(x+4, y+13, z+4, x+10, y+13, z+10, 45)
    mc.setBlocks(x+5, y+13, z+5, x+9, y+13, z+9, 0)
    mc.setBlocks(x+5, y+14, z+5, x+9, y+14, z+9, 45)
    mc.setBlocks(x+6, y+14, z+6, x+8, y+14, z+8, 0)
    mc.setBlocks(x+6, y+15, z+6, x+8, y+15, z+8, 45)
    mc.setBlocks(x+7, y+15, z+7, x+7, y+15, z+7, 45)

def torches():
    mc.setBlocks(x+10, y+1, z+8, x+10, y+1, z+8, 50)
    mc.setBlocks(x+5, y+1, z+8, x+5, y+1, z+8, 50)

def create_house():
    limites_maison()
    murs_intérieurs()
    portes()
    passage_niveau()
    vitres()
    toit()
    torches()

create_house()

```

Ensuite, nous avons créé une **mine** accessible par un tunnel. Dans cette mine médiévale, il est aussi bien possible de prendre une douche froide qu'un bain bouillant dans la lave. Attention aux personnes voulant se relaxer dans l'intimité, la mine est éclairée par des torches.

Voici une photo de cette mine:



Et voici le code:

```
In [4]: from mcpi.minecraft import Minecraft
mc = Minecraft.create()
x, y, z = mc.player.getPos()

def make_gallery():
    mc.setBlocks(x-1, y-10, z-1, x+11, y-15, z+11, 98 )#bloc de base
    mc.setBlocks(x, y-11, z, x+10, y-14, z+10, 0 )#vide
    mc.setBlocks(x, y, z, x, y-11, z, 0)#entree

def make_bassine():
    mc.setBlocks(x+7, y-14, z, x+10, y-14, z+10, 14 )#bloc base bassine
    mc.setBlocks(x+8, y-14, z, x+10, y-14, z+4, 0 )#trou
    mc.setBlocks(x+8, y-14, z+6, x+10, y-14, z+10, 0 )#trou
    mc.setBlock(x+9, y-14, z+7, 8 )#eau
    mc.setBlock(x+9, y-14, z+9, 8 )#eau
    mc.setBlock(x+9, y-14, z+3, 10 )#lave
    mc.setBlock(x+9, y-14, z, 10 )#lave

def torches():
    mc.setBlock(x+9, y-12, z, 50 )
    mc.setBlock(x+9, y-12, z+10, 50 )
    mc.setBlock(x+1, y-12, z+10, 50 )

def sous_terrains():
    make_gallery()
    make_bassine()
    torches()

sous_terrains()
```

```
-----
ConnectionRefusedError                                     Traceback (most recent call last)
<ipython-input-4-cde7b7dc4cf1> in <module>()
      1 from mcpi.minecraft import Minecraft
----> 2 mc = Minecraft.create()
      3 x, y, z = mc.player.getPos()
      4
      5

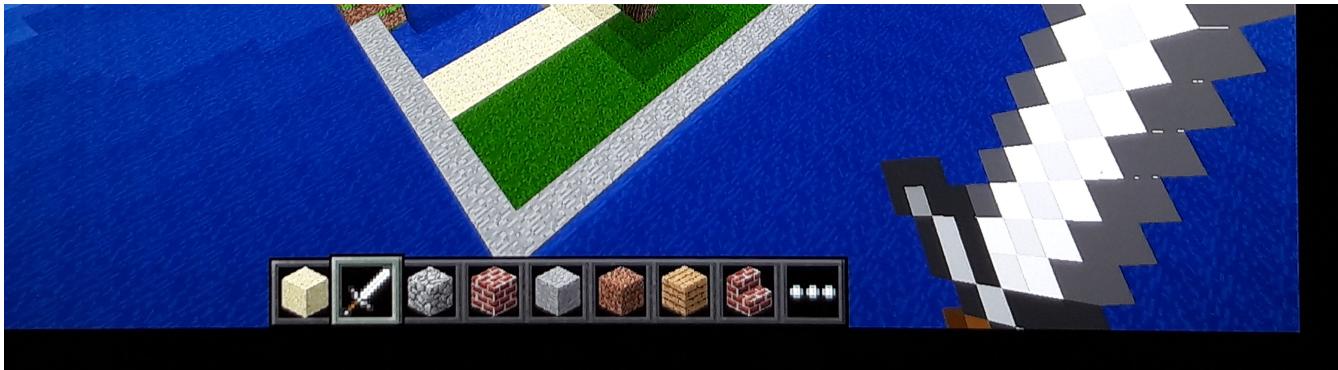
/usr/lib/python3/dist-packages/mcpi/minecraft.py in create(address, port)
  169     @staticmethod
  170     def create(address = "localhost", port = 4711):
--> 171         return Minecraft(Connection(address, port))
  172
  173

/usr/lib/python3/dist-packages/mcpi/connection.py in __init__(self, address, port)
   15     def __init__(self, address, port):
   16         self.socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
--> 17         self.socket.connect((address, port))
   18         self.lastSent = ""
   19
```

ConnectionRefusedError: [Errno 111] Connection refused

Pour finir, nous avons créé une structure extérieure, cette dernière est composée d'une montagne par laquelle jaillit une cascade donnant accès à une plage. Il y a aussi des arbres et une barrière délimitant ce petit paradis extérieur. Voici à quoi cette structure ressemble:





Voici le code:

```
In [5]: from mcpi.minecraft import Minecraft
mc = Minecraft.create()
x, y, z = mc.player.getPos()

def terrain():
    '''Cree une surface simple 15 *15'''
    mc.setBlocks(x, y-1, z, x+300, y-300, z+300, 3)
    mc.setBlocks(x, y, z, x+300, y+150, z+300, 0)
    mc.setBlocks(x, y-1, z, x+300, y-1, z+300, 2)
    mc.setBlocks(x+1, y, z+1, x+300, y, z+300, 0)

def cascade():
    mc.setBlocks(x+9, y, z+10, x+15, y+10, z+15, 2)
    mc.setBlocks(x+11, y+9, z+10, x+13, y+9, z+14, 8) #eau
    mc.setBlocks(x+9, y-1, z+7, x+14, y-1, z+9, 0)
    mc.setBlocks(x+9, y-1, z+5, x+14, y-1, z+6, 12)
    mc.setBlocks(x+8, y-1, z+5, x+7, y-1, z+9, 12)

def arbre():
    mc.setBlocks(x+6, y, z+4, x+6, y+4, z+4, 17) #tronc
    mc.setBlocks(x+4, y+4, z+2, x+8, y+4, z+6, 18)
```