Erasmus+

2Play Learn

Reviving hands-on educational play for learning skills of tomorrow

PROJECT N° 2019-1-UK01-KA201-061466

MODULE 1

Minecraft Pi

DEVELOPED BY D-LEARN & CCSDE

CIVIC idec ΕΚΠΑΙΔΕΥΤΗΡΙΑ ΠΛΑΤΩΝ dlearn scholé Emphasys CENTRE CCS Digital Education

MODULE 1 – Minecraft Pi

MODULE DESCRIPTION

Minecraft Pi is a version of Minecraft, with minimal features, developed for Raspberry Pi. Pi edition is intended as an educational tool for novice programmers, allowing users to enjoy the game and learn programming at the same time.

This resource presents the most important and practical guidelines for Minecraft Pi, such as how to control the player, manually build with blocks and use the Python interface to manipulate the world around you. It is meant for educational purposes and is considered a quick but all-inclusive manual for introducing a new player to Minecraft Pi.

Co-funded by the
Erasmus+ Programme
of the European Union

MODULE 1 – Minecraft Pi

LEARNING OUTCOMES

Once you follow this module you will be able to:

- Access Minecraft Pi and create a new world
- Navigate around Minecraft Pi
- Know how to place and destroy a block, and navigate through different types of blocks in the in-game inventory
- Connect Python to Minecraft Pi
- Use the Python programming interface
- Manipulate blocks using Python scripts
- Make Minecraft interact with the physical world through the Raspberry Pi GPIO

Co-funded by the
Erasmus+ Programme
of the European Union

MODULE 1 – Minecraft Pi

TOPICS

- Introduction to Minecraft Pi basic functions
- Minecraft Pi elements and gameplay
- Controlling Minecraft Pi with Python
- Interaction of Minecraft Pi with the physical world through the Raspberry Pi's GPIO:
  - Connecting LEDs, buttons and switches
  - Create electronic kits to interact with Minecraft Pi

Co-funded by the
Erasmus+ Programme
of the European Union

MODULE 1 – Minecraft Pi

1 Introduction to Minecraft Pi basic functions

LEARNING OUTCOMES

Once you have followed this topic you will be able to:

- Know the basics before running Minecraft Pi
- Run Minecraft Pi on your Play2Learn computer
- Navigate around Minecraft Pi
- Use the controls on your mouse and keyboard

Co-funded by the
Erasmus+ Programme
of the European Union

Playing Minecraft Pi for the first time:

Your Play2Learn computer contains everything that you need to run Minecraft Pi in terms of software. The only peripherals that are needed are a keyboard and your Play2Learn mouse which can be connected through the USB ports of the Play2Learn computer. Make sure your Play2Learn computer is connected to the Internet as well.

Co-funded by the
Erasmus+ Programme
of the European Union

What can be made in Minecraft Pi:

Minecraft Pi is an open-world game where players use blocks that represent different materials to build virtual worlds. You can create anything from a single house to a huge castle and from a small crops field to a big city.

Figure 1 Entering a Minecraft Pi world.

Co-funded by the
Erasmus+ Programme
of the European Union

Minecraft Pi and Python programming language:

Minecraft Pi can be manipulated using Python scripts which interact with various game functions.

The Raspberry Pi edition of Minecraft comes with an API (Application Programming Interface) which allows you to control the game using Python programming.

Python will be used to manipulate blocks and structures, send in-game messages, automated building processes and create fun mini games.

Co-funded by the Erasmus+ Programme of the European Union

Minecraft Pi and multiplayer:

Minecraft Pi supports multiplayer, which means that more than one player can play and interact with each other in the same map.

When several Play2Learn computers are connected via the same Wi-Fi or Ethernet network, the multiplayer mode is enabled, and several users can connect to the same Minecraft world.

Co-funded by the
Erasmus+ Programme
of the European Union

Running Minecraft Pi:

- Double click the desktop icon of Minecraft Pi, or
- Go to **Main menu** (Raspberry Pi logo) → **Games → Minecraft Pi**
- When the game is loaded, click **Start Game → Create New**

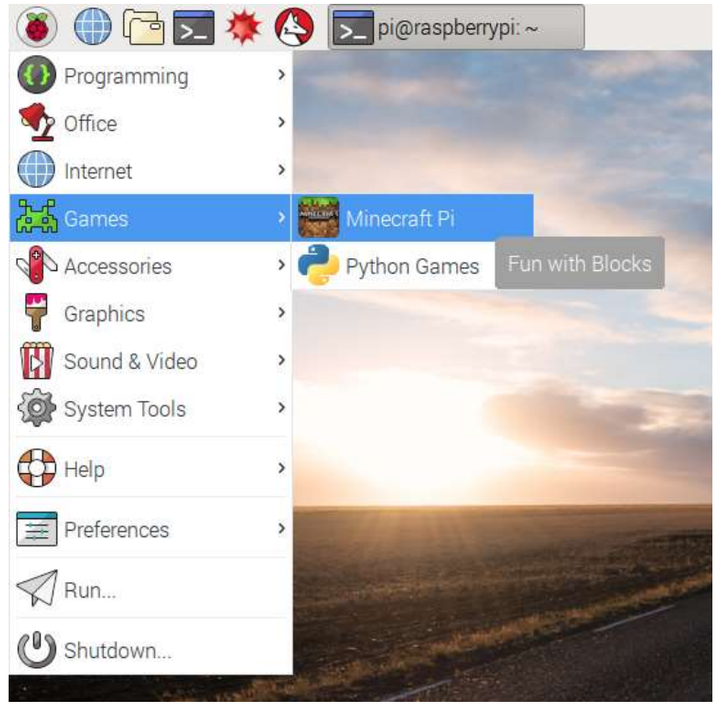

Figure 2 Running Minecraft Pi.

Co-funded by the Erasmus+ Programme of the European Union

Minecraft Pi basic functions:

Basic functions include navigation and control in a Minecraft world as shown in Figure 3:

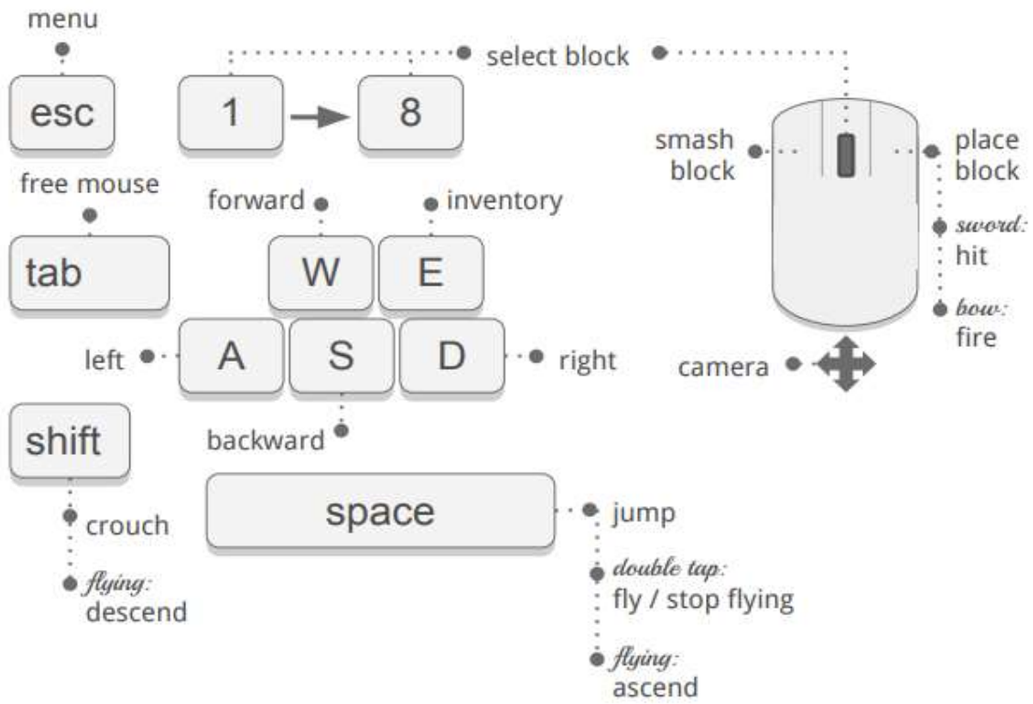

Figure 3 Minecraft controls overview.

Source: https://arghbox.wordpress.com/2013/07/28/minecraft-pi-controls/

Co-funded by the Erasmus+ Programme of the European Union

LEARNING OUTCOMES

Once you have followed this topic you will be able to:

- Use the navigation controls in a Minecraft world
- Know how to place and destroy blocks
- Use the in-game inventory to navigate through different types of blocks

Co-funded by the
Erasmus+ Programme
of the European Union

Entering a Minecraft Pi game:

When first entering the game, the player starts with a sword in hand, which can be used to remove blocks or to dig. On the bottom of the screen there is a toolbox (partial view of the inventory) with different blocks and tools which offers quick access by rolling up or down the mouse wheel.

To navigate in the game:

- Use the mouse to look around, right click to place a block or hit with the sword, left click to smash a block.
- Use the A, W, S, D to move left, forward, backward and right respectively.
- Tap Space bar to make your character jump. Double tap to start ascending (flying). Double tap again to make your character fall to the ground.
- Tap SHIFT to crouch. While flying, tapping SHIFT will descent your character.
- Press E to open the inventory and use your mouse to choose different blocks.
- Tap TAB to release your mouse and use other programs of Raspberry Pi. This will be used frequently when writing scripts.
- Keys 1 to 8 give you quick access to different tools and materials of your inventory toolbar at the bottom of the screen.

Co-funded by the
Erasmus+ Programme
of the European Union

Minecraft Pi Elements:

Blocks:

- Basic units of structure in Minecraft.
- Each block is $1m^3$, except from lava and water blocks which occupy a bigger area.
- Blocks are arranged in a grid and are snapped to it, which means that a block cannot be more than one cell, with some exceptions (e.g., beds).
- Some blocks are opaque while others are transparent. Some emit light. All blocks ignore gravity except sand and snow.

Block IDs:

- Each type of block has a unique Block ID associated with it. Block IDs are needed to manipulate different blocks when writing code in Python.
- For a complete list of block types and block IDs click ***here***.

Co-funded by the Erasmus+ Programme of the European Union

LEARNING OUTCOMES

Once you have followed this topic you will be able to:

- Connect Python to Minecraft Pi
- Use the Python programming interface and learn basic commands
- Manipulate blocks using Python code and scripts
- Have a good understanding of the rest of Minecraft Pi functions that can be manipulated using Python

Co-funded by the
Erasmus+ Programme
of the European Union

Minecraft Pi and Python:

As mentioned earlier, Minecraft Pi has an API that allows a connection with Python programming interface in order to control and manipulate a Minecraft world. Using Python, a user can:

- Get the player's position
- Change or set player's position
- Get the type of block the user wants
- Change a block with another
- Change camera angles
- Post messages to the player
- Build massive structures

Co-funded by the
Erasmus+ Programme
of the European Union

Opening Python programming interface (Figure 4):

- Main Menu → Programming
- Click on Thonny Python IDE

Figure 4 Locating Thonny Python.

Co-funded by the
Erasmus+ Programme
of the European Union

Connecting Python with Minecraft:

- We first need to import Minecraft to our program.
- Then we save the Minecraft game object in a variable called "*mc*" for ease of use.
- In your code, you will use the variable *mc* to refer to different commands as we will see in the following slides.

first.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
```

Figure 5 Connect Python to Minecraft Pi.

Co-funded by the
Erasmus+ Programme
of the European Union

Your first program:

- In Python's programming interface we create a new file by clicking **File → New File**.
- Click **File → Save As** and save it under the name *hello.py*
- Write the code you see on Figure 6.
- Save the file and tap F5 on your keyboard to run your script.

hello.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  mc.postToChat("Hello World!")
```

Figure 6 Sending a message to the player.

Co-funded by the Erasmus+ Programme of the European Union

Finding player's location:

- Finding player's position is necessary before building any structures.
- To do that we use the command `mc.player.getPos()`
- There are two ways of finding player's position:
  - We use a variable named `pos` and we save player's coordinates, as shown in Figure 7.
  - We save player's position is *xyz* coordinates, as shown in Figure 8 (**x** is forward/back, **z** is left/right, and **y** is up/down)

pos.py

```
1 from mcpi.minecraft import minecraft
2
3 mc = Minecraft.create()
4
5 pos = mc.player.getPos()
```

Figure 7 Saving player's position on variable *pos*.

pos.py *

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

Figure 8 Saving player's position on *xyz* coordinates.

Co-funded by the Erasmus+ Programme of the European Union

Teleporting our player:

- Having our player's coordinates also means that these coordinates can be manipulated, meaning that we can teleport our player in various places in a Minecraft world.
- To do that we use the command `mc.player.setPos()`
- Figure 9 shows how to use this command to teleport our player 200 spaces up in the air and then watch it falling to its original position on the ground.

tel.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  x, y, z = mc.player.getPos()
6
7  mc.player.setPos(x, y+200, z)
```

Figure 9 Teleporting our player 200 spaces up in the air.

Co-funded by the Erasmus+ Programme of the European Union

Managing blocks (1/3):

- Apart from finding our player's position and manipulate it, we can use player's position to interact with blocks.
- One of the things we can do is to know the exact block our player is standing on.
- To do that, firstly we find the tile position by using the `mc.getTilePos()` command and then we use the `mc.getBlock(x,y,z)` command, as shown in Figure 10.

tile.py

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

Figure 10 Finding tile position and saving it into coordinates.

Co-funded by the Erasmus+ Programme of the European Union

Managing blocks (2/3):

- Then we generate blocks around our player by using the `mc.setBlock(x,y,z,blockType,blockData)` command.
- Coordinates *xyz* refers to the location in which we generate a block.
- *blockType* refers to the difference block type IDs
- *blockData* refers to extra properties that some block have (e.g., different colors)
- Figure 11 shows how to generate a block of stone (*blockType* is 1) next to our player.

block.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  x, y, z = mc.player.getPos()
6
7  mc.setBlock(x+1, y, z, 1)
```

Figure 11 Generating a block of stone next to the player.

Co-funded by the Erasmus+ Programme of the European Union

Managing blocks (3/4):

As mentioned, some blocks have extra properties. For example:

- Wool (ID-35) → 0: white, 1: orange, 2: magenta, 3: light blue, 4: yellow, etc.
- Wood (ID-17) → 0: oak, 1: spruce, 2: birch, etc.
- Tall grass (ID-31) → 0: shrub, 1: grass, 2: fern
- Torch (ID-50) → 0: pointing east, 1: west, 2: north, 3: south

The whole list of block types and block properties can be found in the following link: https://minecraft.gamepedia.com/Block

Co-funded by the Erasmus+ Programme of the European Union

Managing blocks (4/4):

- Finally, the API allows to generate multiple blocks together so we can create various structures at the click of a button.
- To do that we use the `setBlocks(x1,y1,z1,x2,y2,z2,blockType,blockData)` command.
- It works by specifying two sets of coordinates which the API uses to fill the gap between then with a certain block type.
- Figure 12 shows how to build a 10x10x10 block made of gold.

goldblock.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  gold = 41
6
7  x, y, z = mc.player.getPos()
8
9  mc.setBlocks(x+1, y, z+1, x+11, y+11, z+11, gold)
```

Figure 12 Generating a 10x10x10 block of gold.

Co-funded by the Erasmus+ Programme of the European Union

Special blocks:

- Special blocks withing Minecraft refer to blocks that can interact with the surroundings of a Minecraft world.
- These blocks range from simple mechanisms to flowing lava.
- The whole list of special blocks can be found in the following link: https://minecraft.gamepedia.com/Block
- The following two slides go into more detail about two special blocks that are most commonly used: TNT and Lava blocks.

Co-funded by the Erasmus+ Programme of the European Union

TNT blocks:

- TNT blocks can be used by the player to generate controlled explosions which destroy structures, mountains and fields.
- The player can place a TNT block like any other block, but when right clicking on it a couple times that TNT block is detonated, giving a 4-second window to the player to move away before exploding. Its Block ID is 46.
- Figure 13 shows how to generate TNT blocks using Python code (note that the last digit should always be “1” so the TNT block explodes.

tnt.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  tnt = 46
6
7  mc.setBlock(x, y, z, tnt, 1)
```

Figure 13 Generating TNT blocks using Python programming.

Co-funded by the Erasmus+ Programme of the European Union

Playing with Lava:

- Lava is a light-emitting fluid block that causes fire damage and it spreads is a 3x3 area above it and a 5x5 area below it. Its Block ID is 10.
- Lava can burn flammable structures such as grass and wood, but also the player so we have to be careful not to step on it.
- When Lava cools down, it becomes rock.
- Figure 14 shows how to generate Lave using Python code.

lava.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  lava = 10
6
7  mc.setBlock(x+3, y, z+3, lava)
```

Figure 14 Generating Lava blocks using Python programming.

Co-funded by the Erasmus+ Programme of the European Union

Importing Minecraft modules:

- To ensure our programs run smoothly without crashing the game, we need to import some Minecraft modules at the beginning of our Python script.
- These modules also let us access properties and parameters that are needed in order to manipulate a Minecraft world.
- Figure 15 shows where and how these modules should be imported.

modules.py

```
1  from mcpi.minecraft import minecraft
2  from mcpi.block import block
3  from time import sleep
4
5  mc = Minecraft.create()
6
7  # rest of program
8  # ...
9  # ...
10 # ...
```

Figure 15 Necessary modules imported in a Python script.

Co-funded by the Erasmus+ Programme of the European Union

Recap of Python commands:

- `postToChat("our message")` – communicate with the player in the game;
- `player.getPos()` – get the precise position of a player;
- `player.setPos(x, y, z)` – set (change) the player's position;
- `player.getTilePos()` – get the position of the block where the player currently is;
- `getBlock(x, y, z, blockType, blockData)` – get a block type for a specific position;
- `setBlock(x, y, z, blockType, blockData)` – set (change) a block to a specific block type;
- `setBlocks(x1, y1, z1, x2, y2, z2, blockType, blockData)` set lots of blocks all at the same time by providing 2 sets of x, y, z coordinates.

Co-funded by the Erasmus+ Programme of the European Union

MODULE 1 – Minecraft Pi

4 Minecraft Pi and the physical world

LEARNING OUTCOMES

Once you have followed this topic you will be able to:

- Program buttons to interact with your Minecraft Pi game
- Automate processes, manipulate blocks and create mini games
- Program LEDs to simulate events that happen in game

Co-funded by the
Erasmus+ Programme
of the European Union

Connecting a push button (1/3):

- Required material:
  - 1 x breadboard
  - 1 x push button
  - 1 x 220 Ohm resistor
  - 2 x female-to-male jumper cables
- Connectivity is very simple and is shown in Figure 15.
- Note that 1 jumper cable is connected to a 3.3V pin (red) and the other one is connected to a GPIO input/output pin (pin number 16).

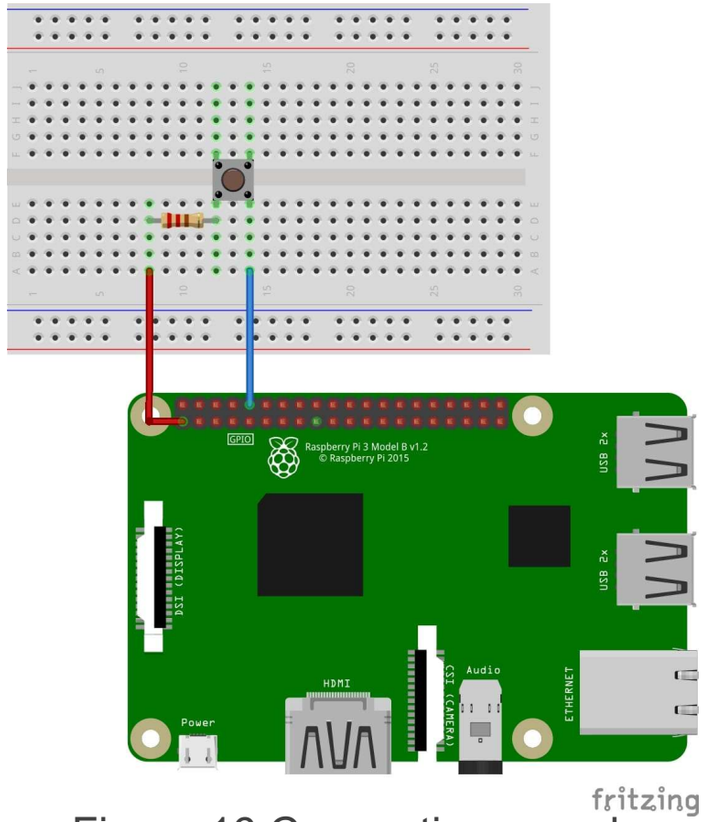

Figure 16 Connecting a push button to Raspberry GPIO. Source: raspberrypihq.com

Co-funded by the Erasmus+ Programme of the European Union

Connecting a push button (2/3):

- Note that you should familiarize yourself with the GPIO pins, which pins to use and their numbering.
- Figure 17 shows the standard Broadcom (BCM) pins names. Numbering is not in numerical order so be aware.

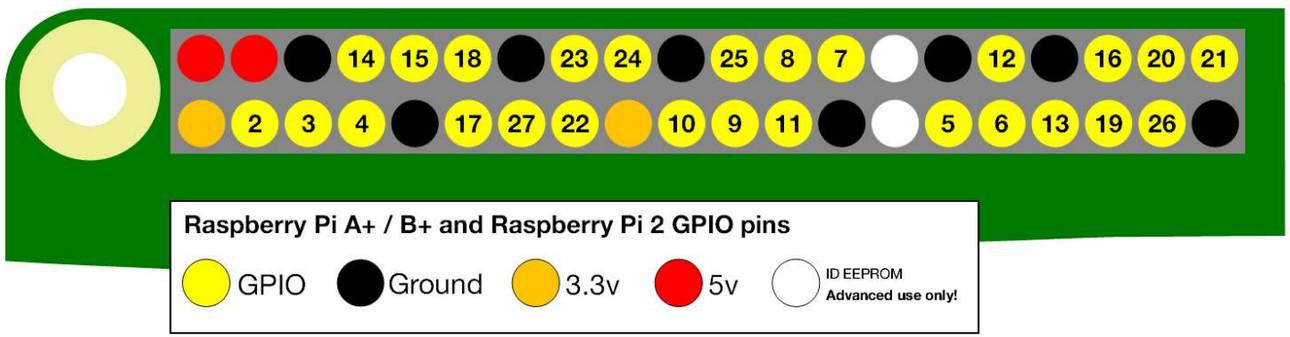

Figure 17 GPIO pins numbering.

Co-funded by the Erasmus+ Programme of the European Union

Connecting a push button (3/3):

- Now that our circuit is ready, we jump to a new Python file to test the button's functionality.
- Figure 18 shows the necessary Python code so we can test our button.
- Save and press F5 to run your script. Now every time you push the button the message "Button works!" should appear in Python's terminal window.
- To stop the script, press Ctrl+C.

button_test.py *

```
1  import RPi.GPIO as GPIO
2  import time
3
4  GPIO.setmode(GPIO.BCM) #tell the Pi what headers to use
5  GPIO.setup(15 GPIO.IN) #tell the Pi pin 15 is an input
6
7  while True:
8      if GPIO.input(15) == True: #look for button press
9          print "Button works!" #log result
10         time.sleep(0.5) #wait 0.5 seconds
```

Figure 18 Python code for testing button functionality.

Co-funded by the Erasmus+ Programme of the European Union

Super mining in Minecraft (1/2):

- Now we will use the previous circuit to interact with Minecraft Pi.
- We will create a program that destroys a block of blocks in a Minecraft world every time you push the button.
- Create a **New File** and **Save** it as *mining.py*.
- Follow the code in Figure 19 (next slide) and see what happens in the Minecraft world when you run your script (F5) and push the button.
- Note that you can manipulate the block types and coordinates as you wish, but do not go too crazy because the Raspberry Pi might struggle.

Co-funded by the
Erasmus+ Programme
of the European Union

Super mining in Minecraft (2/2):

mining.py

```
1  import RPi.GPIO as GPIO
2  import sleep
3  from mcpi.minecraft import Minecraft
4
5  mc = Minecraft.create() #connect Minecraft Pi with Python
6
7  GPIO.setmode(GPIO.BCM) #tell the Pi what headers to use
8  GPIO.setup(15, GPIO.IN) #tell the Pi pin 15 is an input
9
10 while True:
11     if GPIO.input(15) == True #look for button press
12     x, y, z = mc.player.getPos() #read the player's position
13
14     mc.setBlocks(x, y, z, x+10, y+10, z+10, 0) #mine 10 blocks
15     mc.setBlocks(x, y, z, x-10, y-10, z-10, 0) #mine 10 blocks
16
17     time.sleep(0.5) #wait 0.5 seconds
```

Figure 19 Python code for testing button functionality.

Co-funded by the
Erasmus+ Programme
of the European Union

Diamond detector (1/3):

- Required material:
  - 1 x breadboard
  - 1 x LED (any color)
  - 1 x 220 Ohm resistor
  - 2 x female-to-male jumper cables
- Connectivity is very simple and is shown in Figure 20.
- Note that 1 jumper cable is connected to a ground pin (black) and the other one is connected to a GPIO input/output pin (pin number 23).

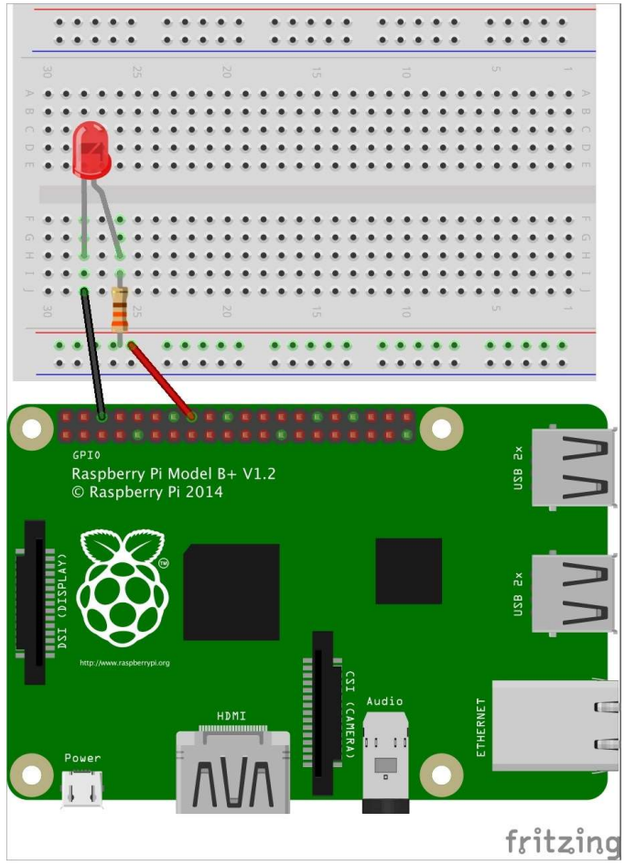

Figure 20 Connecting an LED to GPIO. Source: raspberrypihq.com

Co-funded by the Erasmus+ Programme of the European Union

Diamond detector (2/3):

- Now we will use this new circuit to let Minecraft Pi interact with the physical world.
- We will create a program that will detect diamond blocks as we walk in a Minecraft world.
- Create a **New File** and **Save** it as *detector.py*.
- Follow the code in Figure 21 (next slide), run your script (F5) and move around in a Minecraft world.

Co-funded by the
Erasmus+ Programme
of the European Union

Diamond detector (3/3):

detector.py

```
1  import RPi.GPIO as GPIO
2  import time
3  from mcpi.minecraft import Minecraft
4  from mcpi.blocks import Blocks
5
6  mc = Minecraft.create() #connect Minecraft with Python
7
8  led_pin = 23 #store the GPIO pin number in variable
9
10 GPIO.setmode(GPIO.BCM) #tell the Pi what headers to use
11 GPIO.setup(23, GPIO.OUT) #tell the Pi pin 23 is an output
12
13 while True: #repeat indefinitely
14     x, y, z = mc.playerPos()
15     for i in range(10): #check every block until block 10
16         if mc.getBlock(x, y-i, z) == 56:
17             GPIO.output(led_pin, True) #turn LED on
18             time.sleep(0.5) #wait
19             GPIO.output(led_pin, False) #turn LED off
20             time.sleep(0.5) #wait
```

Figure 21 Python code for creating a diamond detector.

Co-funded by the
Erasmus+ Programme
of the European Union

Extra resources:

Visit the following links for more material on how to program buttons and other hardware to interact with your Minecraft game:

- Adding movement
- Print messages

Co-funded by the
Erasmus+ Programme
of the European Union

EXAMPLES

Example 1 (1/2) – Trap your player between blocks

The code below gets the players tile position, it then calls the Minecraft API's `getBlock()` command to find out the type of block the player is standing on (by subtracting 1 from the y co-ordinate) before using `setBlock()` to create blocks of the same type the player is standing on around him.

(https://www.stuffaboutcode.com/2013/04/minecraft-pi-edition-api-tutorial.html)

For example, if your player is standing on STONE, then STONE blocks will appear around him.

Co-funded by the Erasmus+ Programme of the European Union

EXAMPLES

Example 1 (2/2) – Trap your player between blocks

trap.py *

```
1  from mcpi.minecraft import minecraft
2  from mcpi.block import block
3  from time import sleep
4
5  mc = Minecraft.create()
6
7  playerTilePos = mc.player.getTilePos()
8  blockBelowPlayerType = mc.getBlock(playerTilePos.x, playerTilePos.y - 1, playerTilePos.z)
9  mc.setBlock(playerTilePos.x + 1, playerTilePos.y + 1, playerTilePos.z, blockBelowPlayerType)
10 mc.setBlock(playerTilePos.x, playerTilePos.y + 1, playerTilePos.z + 1, blockBelowPlayerType)
11 mc.setBlock(playerTilePos.x - 1, playerTilePos.y + 1, playerTilePos.z, blockBelowPlayerType)
12 mc.setBlock(playerTilePos.x, playerTilePos.y + 1, playerTilePos.z - 1, blockBelowPlayerType)
13 mc.postToChat("Trapped you")
14 time.sleep(5)
```

Co-funded by the
Erasmus+ Programme
of the European Union

EXAMPLES

Example 2 (1/2) – Build a house

You want to build a house, but do not want to spend hours building blocks one by one. The following code will create a simple house-looking building from scratch. The code is basic and various comment lines are inserted for convenience.

Co-funded by the
Erasmus+ Programme
of the European Union

EXAMPLES

Example 2 (2/2) – Build a house

house.py *

```
1  from mcpi.minecraft import minecraft
2  from mcpi.block import block
3  from time import sleep
4
5  mc = Minecraft.create()
6
7  x, y, z = mc.player.getPos()
8
9  # build walls
10 mc.setBlocks(x+2, y-1, z+2, x+7, y+3, z+8, 5)
11 # remove interior
12 mc.setBlocks(x+3, y, z+3, x+6, y+2, z+7, 0)
13 # make doorway
14 mc.setBlocks(x+2, y, z+5, x+2, y+1, z+5, 0)
15 # make window 1
16 mc.setBlocks(x+4, y+1, z+8, x+5, y+1, z+8, 102)
17 # make window 2
18 mc.setBlocks(x+4, y+1, z+2, x+5, y+1, z+2, 102)
```

Co-funded by the
Erasmus+ Programme
of the European Union

EXAMPLES

Example 3 (1/3) – Build a house with a twist

You learnt to build a house by running your script, but what about building as many houses as you want and wherever you want at the click of the button?

You can assign an arcade-gaming button to build houses every time you hit it. How?

Firstly, you have to connect a button to the GPIO board as you learnt *(make sure you connect to the right pins on the GPIO board. This example uses GPIO24 as an input pin, but you can use whichever best suits your needs and change the code accordingly)*.

Then, follow the script below (Various comment lines are inserted for convenience.):

Co-funded by the
Erasmus+ Programme
of the European Union

EXAMPLES

Example 3 (2/2) – Build a house with a twist

house_twist.py *

```
#connect Python with Raspberry Pi GPIO board
import RPi.GPIO as GPIO

from mcpi.minecraft import minecraft
from mcpi.block import block
from time import sleep

mc = Minecraft.create()

GPIO.setmode(GPIO.BCM) #set mode for GPIO
GPIO.setup(24, GPIO.IN) #set input pin to GPIO24

while True:
    if GPIO.input(24) == True:
       x,y,z = mc.player.getPos() #get players position
       mc.setBlocks(x+2,y-1,z+2,x+7,y+3,z+8, 5) #make shell
       mc.setBlocks(x+3,y,z+3,x+6,y+2,z+7, 0) #remove inside
       mc.setBlocks(x+2,y,z+5,x+2,y+1,z+5, 0) #make doorway
       mc.setBlocks(x+4,y+1,z+8,x+5,y+1,z+8, 102) #make window 1
       mc.setBlocks(x+4,y+1,z+2,x+5,y+1,z+2, 102) #make window 2
       mc.setBlocks(x+7,y+1,z+4,x+7,y+1,z+6, 102) #make window 3
       print ("House is built")
       time.sleep(0.1) #wait 0.1 sec
```

Co-funded by the
Erasmus+ Programme
of the European Union

CONCLUSION

If you followed this resource with your Raspberry Pi, you are expected to:

- Access Minecraft Pi and create a new world.
- Navigate around Minecraft Pi using the movement controls on your keyboard.
- Know how to place and destroy a block and navigate through different types of blocks in the in-game inventory.
- Connect Python to Minecraft Pi.
- Use Python programming interface.
- Manipulate blocks using Python code and scripts.
- Have a good understanding of the rest of Minecraft Pi functions.
- Make Minecraft interact with the outside world with the use of buttons and LEDs.

Co-funded by the
Erasmus+ Programme
of the European Union

GLOSSARY

Term	Description
Raspberry Pi	Raspberry Pi is a credit card sized, fully functional computer which operates on Raspberry Pi OS.
Minecraft	Minecraft is an open-world educational game where players can build their own virtual worlds with blocks that represents different material.
Raspberry Pi OS	The operating system for Raspberry Pi.
Python	Object-oriented programming language that will be used to build things automatically in Minecraft.
Function	They are text files that contain a command for each action to be completed in the game or in your computer in general.
Code	The piece of instructions that you can set for realization of actions in the game.
Terminal window	A program that is offered by the operating system and is used to execute scripts.
Input command	The command you give the game to be processed (ex. To turn on the light once the block has been created).
Output command	The result of processed command that you have set (ex. The turning on of the light once a block is created).

Co-funded by the Erasmus+ Programme of the European Union

REFERENCES

- Richardson, C., (2013), Minecraft Pi book, retrieved from https://arghbox.files.wordpress.com/2013/06/minecraftbook.pdf
- Minecraft controls, retrieved from https://arghbox.wordpress.com/2013/07/28/minecraft-pi-controls/
- Block ID numbers, retrieved from https://www.raspberrypi-spy.co.uk/2014/09/raspberry-pi-minecraft-block-id-number-reference/

- O'Hanlon, (2013), Minecraft: Pi Edition - API Tutoria, retrieved from https://www.stuffaboutcode.com/2013/04/minecraft-pi-edition-api-tutorial.html
- Special blocks in Minecraft, retrieved from https://minecraft.gamepedia.com/Block
- Minecraft Wiki, retrieved from https://minecraft.gamepedia.com/Pi_Edition

Co-funded by the Erasmus+ Programme of the European Union

EXTRA RESOURCES

- Minecraft API: https://www.stuffaboutcode.com/p/minecraft-api-reference.html
- Raspberry Pi: https://www.stuffaboutcode.com/p/raspberry-pi.html
- Minecraft Wiki: https://minecraft.gamepedia.com/Pi_Edition
- Manhattan project in Minecraft: https://www.stuffaboutcode.com/2013/04/minecraft-pi-edition-manhattan-stroll.html
- Massive Analogue Clock: https://www.stuffaboutcode.com/2013/02/raspberry-pi-minecraft-analogue-clock.html
- Planetary gravity simulation: https://www.stuffaboutcode.com/2013/03/raspberry-pi-minecraft-planetary.html
- Coding Tips: http://www.laschina.org/wp-content/uploads/2017/09/Minecraft-Coding-Tips.pdf

Co-funded by the Erasmus+ Programme of the European Union

Erasmus+

2Play2Learn

Reviving hands-on educational play for learning skills of tomorrow

PROJECT N° 2019-1-UK01-KA201-061466

CIVIC idec ΕΚΠΑΙΔΕΥΤΗΡΙΑ ΠΛΑΤΩΝ dlearn EUROPEAN DIGITAL LEARNING NETWORK scholé Emphasys CENTRE CCS Digital Education