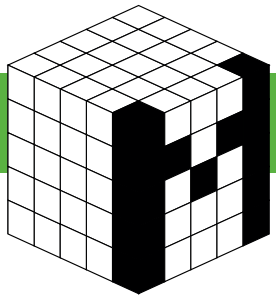


Block Deck 2

Craft commonly-used blocks at the push of a button without heading for the inventory.

60 mins

1. Component list

In addition to your Raspberry Pi and breadboard, you will need:

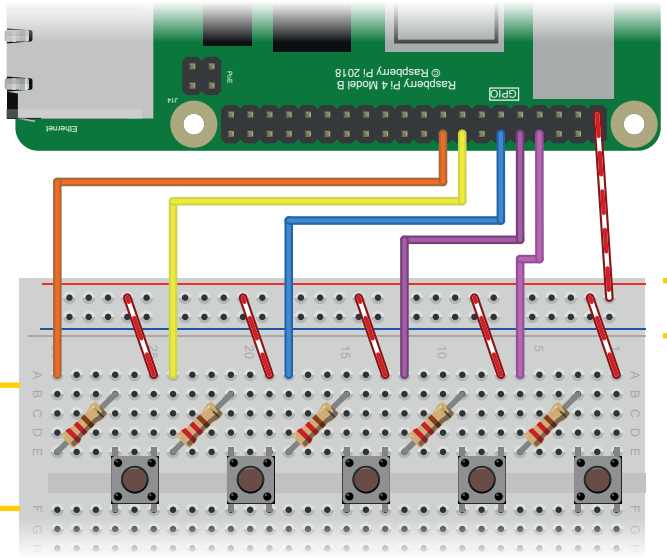
6 x male to female jumpers

5 x male to male jumper

5 x momentary push button

5 x 220 ohm resistor

2. Build your circuit



When a button is pressed, current travels through the button and resistor and to a GPIO pin

The +3.3v pin on the Raspberry Pi is connected to the positive rail on the breadboard. It is then distributed to the push-switches

3. Choosing our blocks

Our circuit has five push-switches, so we can define five blocks to craft. Each type of block has a reference number. We will use these reference numbers in our program. Here are some of the more common ones:

0 Air	10 Lava flowing	45 Brick block
1 Stone	11 Lava stationary	47 TNT
2 Grass	12 Sand	49 Obsidian
3 Dirt	14 Gold ore	50 Torch
5 Wood planks	15 Iron ore	51 Fire
6 Sapling	16 Coal ore	54 Chest
7 Bedrock	41 Gold block	79 Ice
8 Water flowing	42 Iron block	85 Fence
9 Water Stationary	44 Stone slab	107 Fence gate

4. Coding

```
1.  # Import Minecraft Library
2.  import mcpi.minecraft as minecraft
3.  import mcpi.block as block
4.
5.  # Import the GPIO Libraries
6.  import RPi.GPIO as GPIO
7.
8.  # Import the time Library
9.  import time
10.
11. # Create an object and link it to Minecraft
12. mc = minecraft.Minecraft.create()
13.
14. # Set which pin numbering to use and turn off warnings
15. GPIO.setmode(GPIO.BOARD)
16. GPIO.setwarnings(False)
17.
18. # Store pins in a list
19. buttonPins = [18, 16, 12, 10, 8]
20.
21. # Set up buttons
22. for i in buttonPins:
23.     GPIO.setup(i, GPIO.IN, pull_up_down = GPIO.PUD_DOWN)
24.
25. # Make a new block
26. def makeBlock(blockType):
27.     x, y, z = mc.player.getPos()
28.     mc.setBlock(x, y, z, blockType)
29.     time.sleep(0.5)
30.
31. # Main program
32. while True:
33.     # If the pin in the first list index is pressed...
34.     if (GPIO.input(buttonPins[0])):
35.         # Call makeBlock, pass block 1 (stone) to it
36.         makeBlock(1)
37.
38.     # If the pin in the second list index is pressed...
39.     elif (GPIO.input(buttonPins[1])):
40.         # Call makeBlock, pass block 6 (sapling) to it
41.         makeBlock(6)
42.
43.     # If the pin in the third list index is pressed...
44.     elif (GPIO.input(buttonPins[2])):
45.         # Call makeBlock, pass block 14 (gold ore) to it
46.         makeBlock(14)
47.
48.     # If the pin in the fourth list index is pressed...
49.     elif (GPIO.input(buttonPins[3])):
50.         # Call makeBlock, pass block 47 (TNT) to it
51.         makeBlock(47)
52.
53.     # If the pin in the fourth list index is pressed...
54.     elif (GPIO.input(buttonPins[4])):
55.         # Call makeBlock, pass block 85 (fence) to it
56.         makeBlock(85)
```

Import libraries which:

- **Give access to Minecraft and its blocks**
- **Give access to the GPIO pins on the Raspberry Pi**
- **Allow us to add a delay**

Our GPIO pins are not numbered sequentially, so it's easier to store them in a list and iterate through them

And this is where we iterate through the list to set them up as inputs

This procedure takes a block ID as an argument. It then gets the player's position, and adds the block at the same position. The sleep() command prevents multiple accidental button presses

Each if / elif statement checks for input on a specific GPIO pin, according to its position in the list of pins.

If the GPIO registers a button being pressed, the makeBlock() procedure is called with the block ID passed as an argument.

The elif statements are nearly identical. Save time by copying and pasting, then changing the buttonPins index and makeBlock argument