A Cornish Landscape in Minecraft

A feature of Cornwall are the remains of the mining industry. In quite a few places around the county it's not difficult to see more than just a couple within your view. This exercise aims to reproduce these engine houses complete with the mine itself for you to walk down and retrieve any ore that is there.

Reading the Instructions

To help you along these instructions are formatted a certain way. Anything that we know tends not to be read but is important to do is highlighted in bold text. A lot of questions concerning difficulties with the programming are from not reading the instructions properly.

The code that is to be typed into the Python window will be in a different font and blue in colour. **These two lines are not code so please don't ask "Do I type this in?"**

```
# A hash at the front is a comment ignored by Python. It is a human message!
This line shows what code looks like. This is not code.
```

Minecraft Pi

The version of Minecraft that comes with the Raspberry Pi is based on the original Minecraft Pocket Edition. It is in Creative mode only, so no mobs or animals. But the most important aspect of the Minecraft Pi Edition is the programming feature. The intention of Minecraft Pi is that it is used as a way of providing an environment where the results of your programming can be rewarded within the Minecraft game itself. Information can be gathered from the world and the program can also interact with the game too.

Programming Environment

To program into the game you shall be using the programming language Python. The code can be written and run by various methods within a Raspberry Pi but we shall be using the Python3 IDE. So let's make a start.

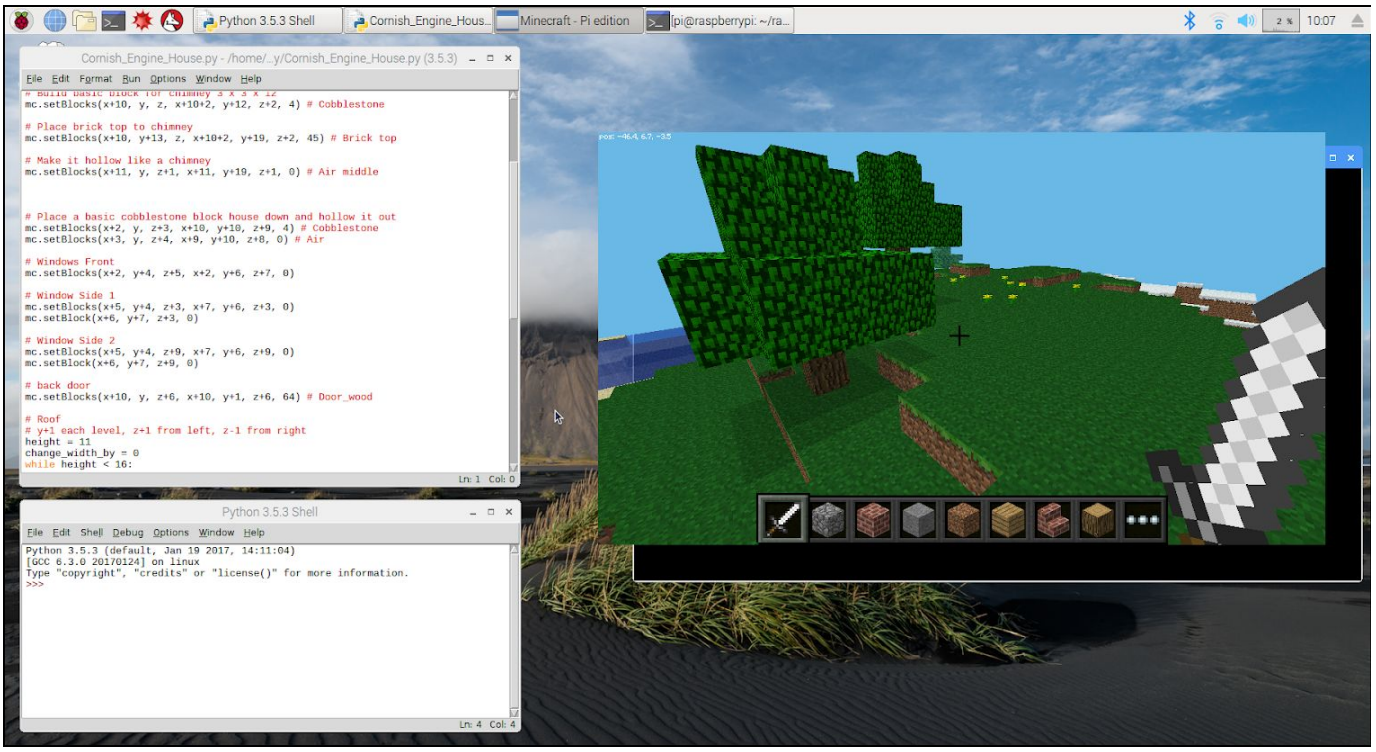
Click on the Menu Raspberry icon to the left of the screen. Under Programming click on Python3 IDE.

The Python Shell will now open. You can program in Python using this but the code will not be saved for later editing. **In the Shell click on File > New File and a second window will open.** Organise the two windows so that they do not take up all the space on the screen. The Shell does not need to be very big as this will just show you any errors in your code when it runs. The remaining space is needed to have the Minecraft window. It is better if that is not too large as it tends to slow the Pi down if too big. Just a warning.

Minecraft Pi

Now open the game. Go to the Menu > Games > Minecraft. Start a new game and open a New World.

Place the game on the screen and adjust the window so that all three windows can be seen at the same time.



The game runs the same way as the PC version. Moving around is by using WASD keys, looking around by the mouse. Jump with the space bar, double click space bar to fly up, shift to come down, double click space bar to stop flying. Inventory of blocks is E, select with the mouse. Change blocks in the hand with the number keys. Left click to mine a block, right click to place a block.

An important feature is to be able to leave the game whilst it is still running. To do this click Tab. The mouse is now back in the screen control. To get back in the game click the mouse over the game. You will need to do this to interact with the game when coding.

Let's Start Programming

The programming is done in the new Python3 IDE window. First of all we will write a short program to create a flat world to play around in. We will create the Cornish landscape later on. This first program will also show you the speed that the Python programming can change the Minecraft world.

First of all save the file into the Documents folder calling it clearGround.py. Now type in these first lines.

```
from mcpi import minecraft
from mcpi import block

mc = minecraft.Minecraft.create()

mc.setBlocks(-200, -1, -200, 200, -1, 200, block.GRASS.id)
mc.setBlocks(-200, -2, -200, 200, -50, 200, block.COBBLESTONE.id)
mc.setBlocks(-200, 0, -200, 200, 50, 200, block.AIR.id)
```

The first lines import into your program the information about the game and blocks. The third line uses some of this to create a link from Python into the game and calls it mc. It can called anything you like, like gamelink. But mc is nice and short. The last three lines use the mc variable and then set three types of blocks as layers into the game. Cobblestone under the ground topped by a layer of grass and then air above it. More about these lines later on.

Save this code using Ctrl + S keys or File > Save. Now make sure your world game is running. Run the code by clicking Run > Run Module or F5. The world should very quickly flatten.

It Didn't Work

Did you save the file? Check the Shell window it might give you some hints as to what was wrong. Check for spelling errors in the code. Capital letters are important. M is not the same as m.

Let's Start Building

We can now use the previous code to start a new program. Save As that file as a new one called cornishEngineHouse.py.

Delete the last three lines of code leaving just two.

We will start by finding the position of the block the player is standing on. Again using the mc link like before. Anytime we want to call into the game you just use mc. This line gets the three coordinates of the player and calls them three separate variables x, y and z.

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

Just to show that you have the information we will put the values onto the Minecraft screen using the game talk feature. Save the file and run (Ctrl + S then F5).

```
mc.postToChat(x)
mc.postToChat(y)
mc.postToChat(z)
```

It Didn't Work

Did you save the file? Don't forget the () at the end of the first new line. Check the Shell for errors.

Place a Starting Block

You are going to be building quite a complex structure and it helps to have somewhere to start from. The Engine House will be built a few blocks away from where the player is standing. Each time you add a few new lines of code a new piece of building will appear. By standing in exactly the same place each time the pieces will be added in the correct place. But by moving around though you can also end up with all kinds of odd pieces of building laying around.

Delete the last three lines of the previous code again. To clear the world before you used a setBlocks command. This places a block of blocks into the world. The code below places just a single block. The difference is just the single s, setBlock or setBlocks. So now add the following:

```
# Place a standing on block of Lapis Lazuli
mc.setBlock(x, y-1, z, block.LAPIS_LAZULI.id)
```

Remember the # line is a comment message for you to read and is ignored by the code. A lot of this code will start to look the same so the comments make it easy to find those pieces of code that might not be working correctly. Save and run the code and then look down in the game. By looking down and placing the crosshairs on the Lapis Lazuli block the Engine House will always be built in the same place.

Build the Chimney

The code now needs to be added for the chimney stack. Engine house were built with cheap stone for the base but once they got to a certain height the stone was too heavy so more expensive, but lighter, bricks were used. The Minecraft stack will do the same.

```
# Build basic block for chimney 3 blocks x 3 wide x 12 high in Cobblestone
mc.setBlocks(x+10, y, z, x+10+2, y+12, z+2, block.COBBLESTONE.id)
```

The code above works like this. There are two sets of coordinates used marking the bottom left of the stack and the top right. The code fills in between these coordinates with all the same blocks.

The first x coordinate has 10 added to it so the block is placed 10 blocks away from the player. The second x has an additional 2 more added to it making the stack 3 blocks wide at positions 10, 11 and 12. The z positions are the same just in the other direction. The y coordinate is the height of the stack from the ground up to 12 blocks high. The next lines below then add a brick top onto the cobblestone

```
# Place brick top to chimney
mc.setBlocks(x+10, y+13, z, x+10+2, y+19, z+2, block.BRICK.id)

# Make it hollow like a chimney
mc.setBlocks(x+11, y, z+1, x+11, y+19, z+1, block.AIR.id)
```

Now go into the game and look at your chimney. Fly up and check that it is hollow then go back to the Lapis Lazuli starting block, if that's what you are doing!

If you like this code could be saved as it is and used later on to spawn random chimneys around your Minecraft world. Just like the real thing! Save As the file as chimney.py and then reopen the code again and carry on.

It Didn't Work

Check for the s at the end of setBlocks. Capital letters where they should be. Closing brackets ().

Adding the Main Building

The actual Engine House is built the same way. A block of blocks is place down and hollowed out. Add these lines to your code:

```
# Place a basic cobblestone block house down and hollow it out
mc.setBlocks(x+2, y, z+3, x+10, y+10, z+9, block.COBBLESTONE.id)
mc.setBlocks(x+3, y, z+4, x+9, y+10, z+8, block.AIR.id)
```

Save, run and go take a look. Knock a hole in the side to see or fly up.

Now add the windows and a door. The door will not open you have to knock it out of the way.

```
# Window Front
mc.setBlocks(x+2, y+4, z+5, x+2, y+6, z+7, block.AIR.id)

# Window Side 1
mc.setBlocks(x+5, y+4, z+3, x+7, y+6, z+3, block.AIR.id)
mc.setBlock(x+6, y+7, z+3, block.AIR.id)

# Window Side 2
mc.setBlocks(x+5, y+4, z+9, x+7, y+6, z+9, block.AIR.id)
mc.setBlock(x+6, y+7, z+9, block.AIR.id)

# Back door
mc.setBlocks(x+10, y, z+6, x+10, y+1, z+6, block.DOOR_WOOD.id)
```

Save and run again. Go inside and take a look.

The Roof

This time the code is slightly different. You will use code to change the width and height of the roof as it builds up. The code uses two new variables simply called height and change_width_by. These are added and subtracted from the coordinates to place the blocks. The roof starts at a height of 11 blocks up adding it to the top of the building of course. The roof also overhangs the building by one block.

The code works like this. The height starts at 11. The code checks it is less than 16 which it is. So the code after the while line runs. It sets the blocks by adding the height to y. It then adds the change_width_by value to z. It starts at zero so there is no change. Then the variables have one added to their values becoming 12 and 1. The code keeps checking the height is less than 16 when it is not the code breaks out of the while block ready to do what follows. Which at the moment is nothing.

```
# Add the Roof
height = 11
change_width_by = 0

while height < 16:
      # Next setBlocks line is all one line not two
      mc.setBlocks(x+2, y+height, z+2+change_width_by, x+10, y+height,
z+10-change_width_by, block.STONE_BRICK.id)
      height += 1
      change_width_by += 1
```

Save and run. If there is an error make sure that there is an indent after the while line. The IDE should do it automatically if everything is typed correctly.

Didn’t Work

You should now be getting the idea. The main one here is the indent after the while line. All code indented after a colon (:) indicates that the code belongs to the while block.

Adding the Balance Beam and Water Shaft

These next lines simply add the wooden beam to the front and a shaft with water in it.

```
# Shaft for water extraction
mc.setBlocks(x-1, y-1, z+5, x-3, y-6, z+7, block.AIR.id)
mc.setBlocks(x-1, y-7, z+5, x-3, y-20, z+7, block.WATER.id)

# Beam Engine
mc.setBlocks(x+4, y+8, z+6, x-2, y+8, z+6, block.WOOD_PLANKS.id)
mc.setBlocks(x-2, y-6, z+6, x-2, y+7, z+6, block.FENCE_GATE.id)
```

The Engine House is now complete. Go take a look. If you walk around your flat world a bit by running the code again and again you can place loads of Engine Houses down. And spare chimneys if you saved the code earlier.

Go Mining

What's the point of an Engine House without a mine to go explore?

This code will add a stepped mine down from the centre of the Engine House.

Here we use another code practice of checking if something is true. In this case if the depth when divided by 4 gives no remainder (this is the modulus) then the code runs after it. e.g 8 divided by 4 leaves 0. So the modulus is the same as (==) zero.

If when the depth is divided by 4 it leaves a remainder then the code is ignored. 9 divided by leaves one. The modulus is not the same as zero so it is not true and does not run.

The code also uses new variables again. The adit_length is the length of the tunnels that run off from the shaft to the left and right.

```
# Lets start mining!

depth = 1
step = 0
adit_length = 0

while depth < 100:
    mc.setBlocks(x+4+step, y-depth, z+5, x+7+step, y-depth, z+7, 0)
    if depth%4 == 0:
        for adit_length in range(0,50):
            mc.setBlocks(x+4+step, y-depth, z+7+adit_length, x+7+step,
y+1-depth, z+6+adit_length, 0)
            mc.setBlocks(x+4+step, y-depth, z+4-adit_length, x+7+step,
y+1-depth, z+5-adit_length, 0)
    depth += 1
    step += 1
```

Save and run. Pick up a torch from the inventory and go exploring.

Make a Cornish Landscape

So now the code is finished. Close the Minecraft world you have been using and start a new one. Find somewhere to put your Engine House (and spare chimneys) around the world. Congratulations. Take a picture of your world with your phone.