RASPBERRY PI
Sense HAT

Scope of this session

Introduction

- The Sense HAT, which is a fundamental part of the Astro Pi mission, allows your Raspberry Pi to sense the world around it.

What we will need

- **Hardware**
  - *Raspberry Pi*
  - *Sense HAT*
- **Software**
  - *You will need the latest version of Raspbian which already includes the following software packages: Python 3 and Sense HAT for Python 3*
  - *Then import all libraries*

Sense Hat Installation

- Ensure your APT package list is up-to-date:

```
sudo apt-get update
```

- Next, install the sense-hat package which will ensure the kernel is up-to-date and install the necessary libraries and programs:

```
sudo apt-get install sense-hat
```

- Finally, a reboot may be required if I2C was disabled

or the kernel was not up-to-date prior to the install:

```
sudo reboot
```

What is a Sense HAT?

- The Sense HAT is an add-on board for the Raspberry Pi, made especially for the Astro Pi competition

- The board allows you to make measurements of temperature, humidity, pressure, and orientation, and to output information using its built-in LED matrix

- The LED matrix is an RGB565 framebuffer with the id "RPi-Sense FB".

Displaying text on Sense HAT

- Make sure you have the following lines of code in your Python program to set up your connection with the Sense HAT. There is no need to add them more than once.

```
 from sense_hat import SenseHat
sense = SenseHat()
```

- Add this code to display a message on the Sense HAT’s LED matrix

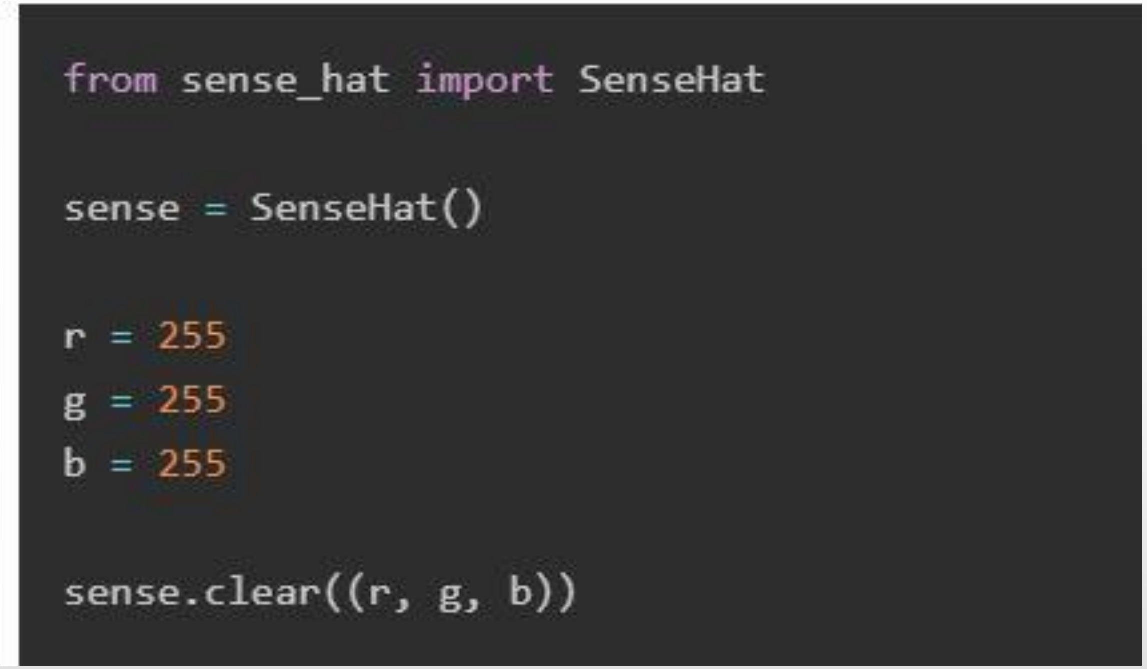
```
sense.show_message("Hello world")
```

The message “Hello world” will now scroll across the LED screen.

Now, Display the text “Raspberry Pi is awesome” on your Sense HAT’s LED display.

Displaying A Color On The Sense HAT

- In a Python file, type in the following code:



```
from sense_hat import SenseHat

sense = SenseHat()

r = 255
g = 255
b = 255

sense.clear((r, g, b))
```

- Save and run your code. The LED matrix will then go bright white.

Displaying A Color On The Sense HAT

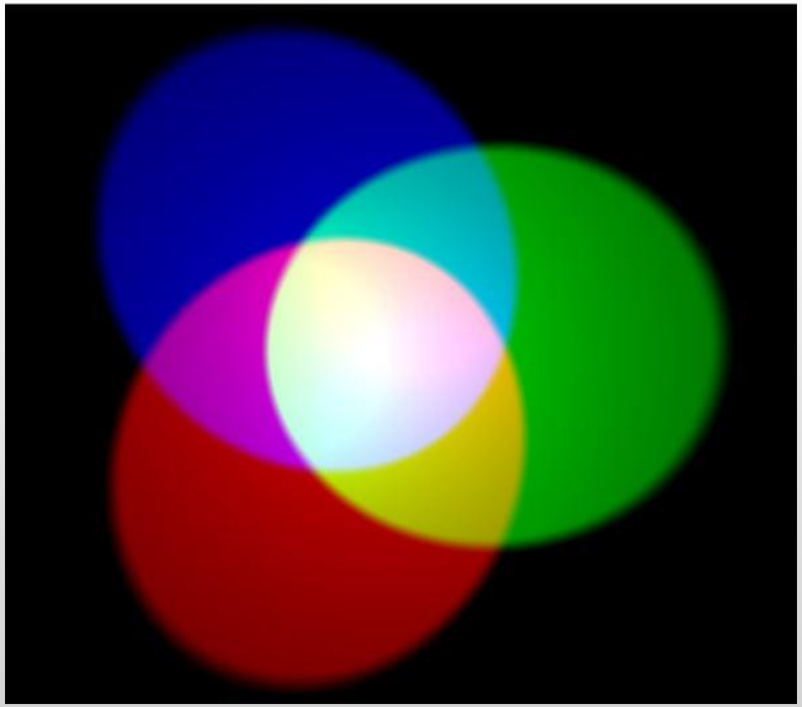
- You can also define all three RGB values of a color using a single line of code: red=(255,0,0)

- Change the value of one of the colors, then run the code again. What do you see?

- Which other colors can you make?

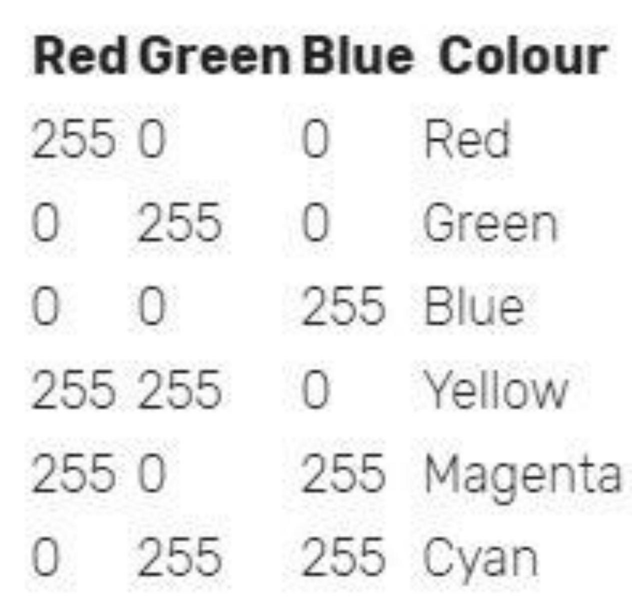
Representing colors with numbers

- Humans see color because of special cells in our eyes. These cells are called *cones*. We have three types of cone cells, and each type detects either red, blue, or green light. Therefore all the colors that we see are just mixtures of the colors red, blue, and green.



Representing colors with numbers

- Computers store everything as 1s and 0s.
- These 1s and 0s are often organized into sets of 8, called **bytes**.
- A single byte can represent any number from 0 up to 255 .
- Here’s a table showing some color values:



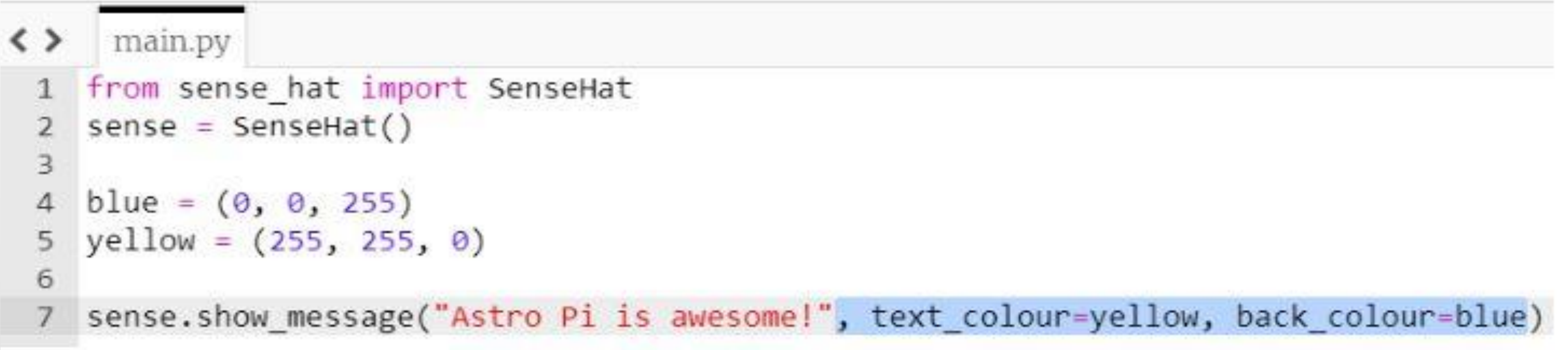
Red	Green	Blue	Colour
255	0	0	Red
0	255	0	Green
0	0	255	Blue
255	255	0	Yellow
255	0	255	Magenta
0	255	255	Cyan

RGB Colors

- RGB color values are supported in all browsers.
- An RGB color value is specified with: rgb(red, green, blue).
- Each parameter (red, green, blue) defines the intensity of the color as an integer between 0 and 255.
- For example, rgb(0, 0, 255) is rendered as blue, because the blue parameter is set to its highest value (255) and the others are set to 0.

Representing Colors With Numbers

- back_color: alters the color of the background and works in the same way as text_color.



```python
from sense_hat import SenseHat
sense = SenseHat()

blue = (0, 0, 255)
yellow = (255, 255, 0)

sense.show_message("Astro Pi is awesome!", text_colour=yellow, back_colour=blue)
```

Displaying a single character

- Add this code to display a single letter “A” on the LED matrix:

```
sense.show_letter(A")
```

Display the letter “Z” on your Sense HAT’s LED display.

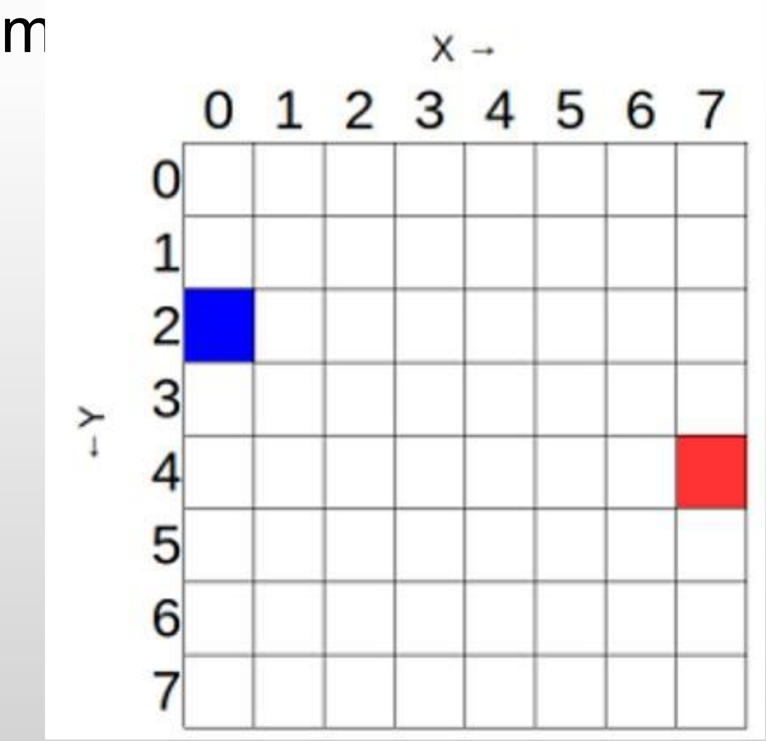
```
from sense_hat import SenseHat
sense = SenseHat()
sense.show_letter("Z")
```

Display the letter “J” in red on a white background.

Displaying Images

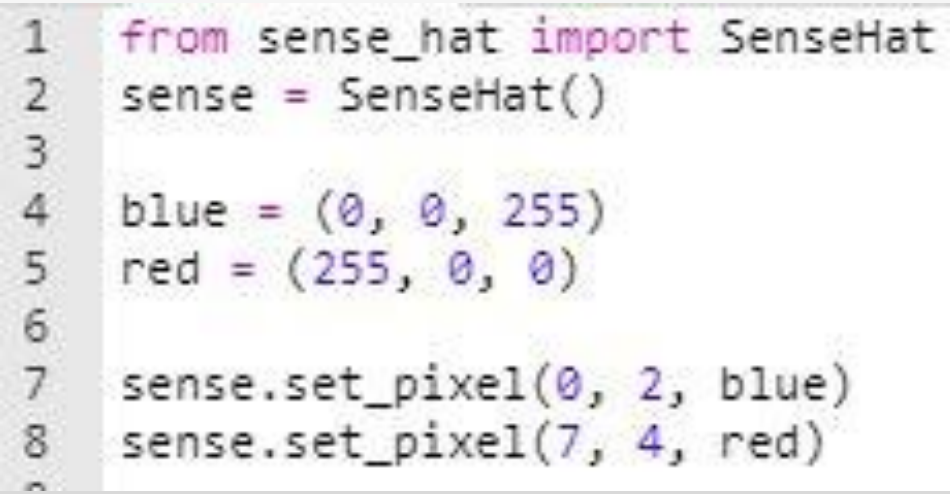
- The LED matrix can display more than just text!
- We can control each LED individually to create an image.
- Sense HAT LED matrix coordinates
- The Sense HAT’s LED matrix uses a coordinate system with an x- and a y-axis.

  - The blue pixel is at coordinates 0, 2.
    The red pixel is at coordinates 7, 4.



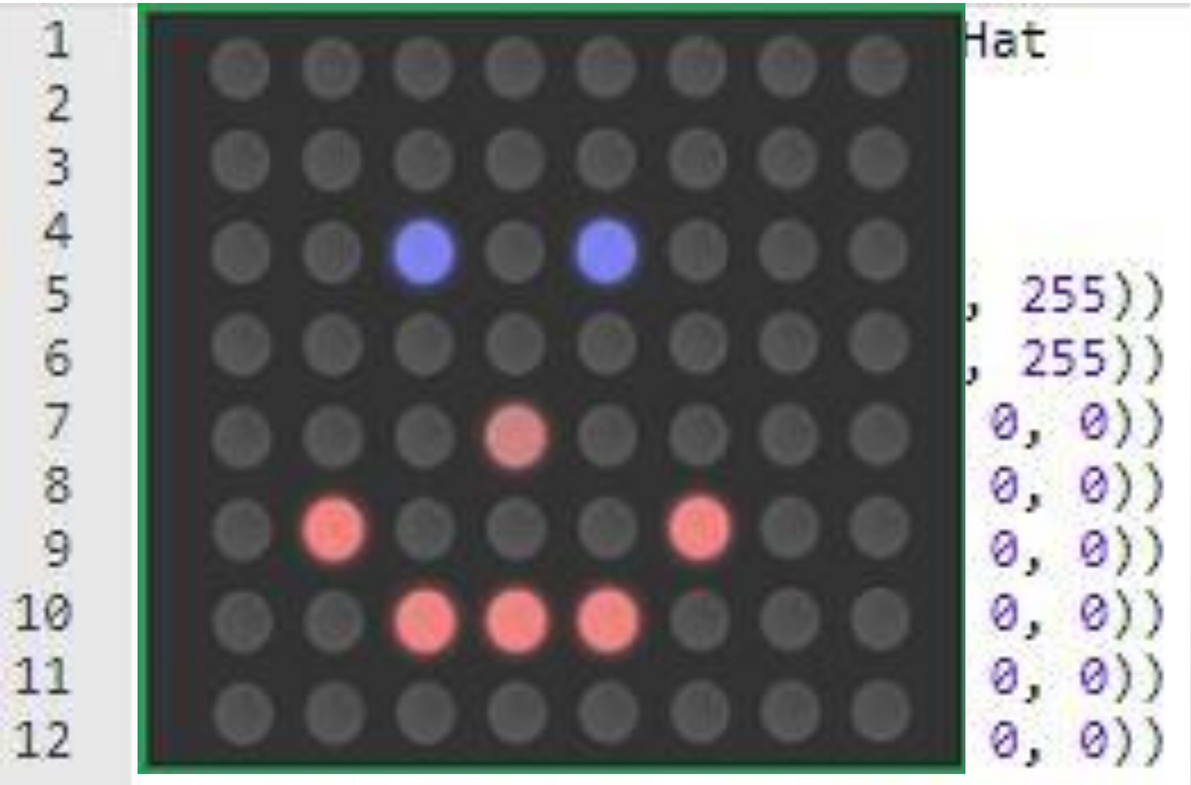
Displaying Images

- You can set pixels (LEDs) individually using the set_pixel() method.
- To replicate the diagram above, you would enter a program like this.



```
from sense_hat import SenseHat
sense = SenseHat()

blue = (0, 0, 255)
red = (255, 0, 0)

sense.set_pixel(0, 2, blue)
sense.set_pixel(7, 4, red)
```

Can You Guess What This Code Creates?



```
Hat



, 255))
, 255))
0, 0))
0, 0))
0, 0))
0, 0))
0, 0))
0, 0))
```

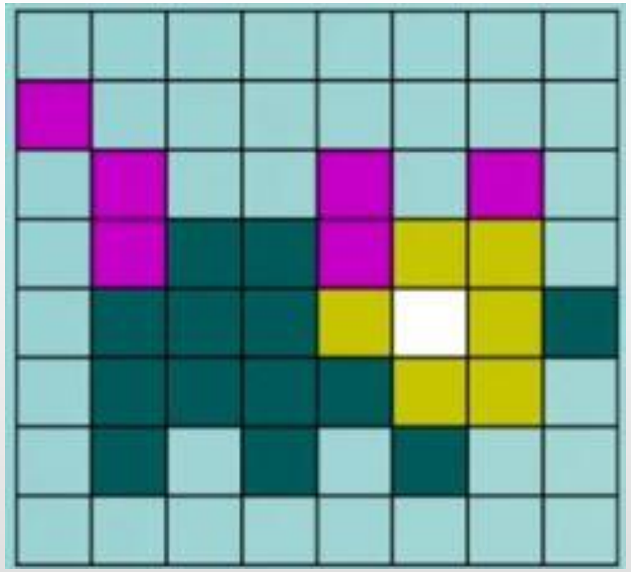
Setting Multiple Pixels

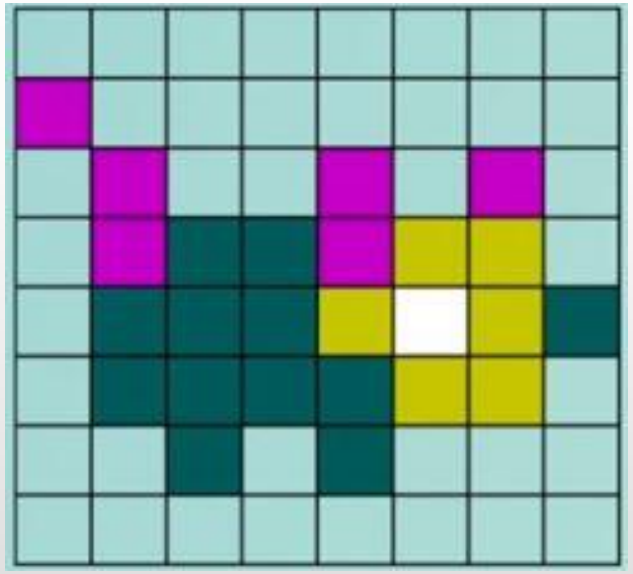
- To change all the pixels in one go with the set_pixels command.
- Use the set_pixels method to display an image on the LED matrix.

- For example you could draw a pet on the LED Matrix

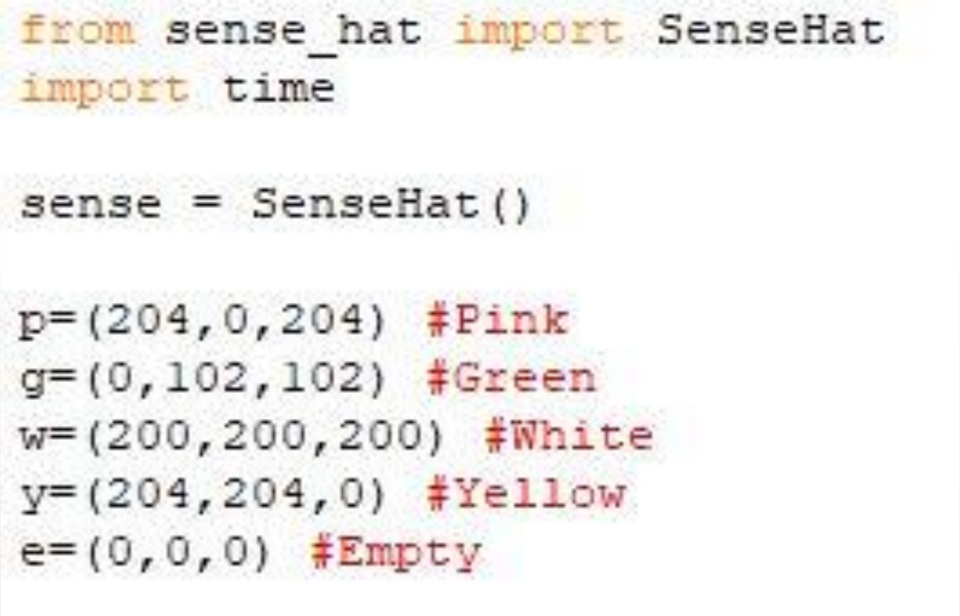
Moving Pixel Avatar Pet

- First draw and design your pet





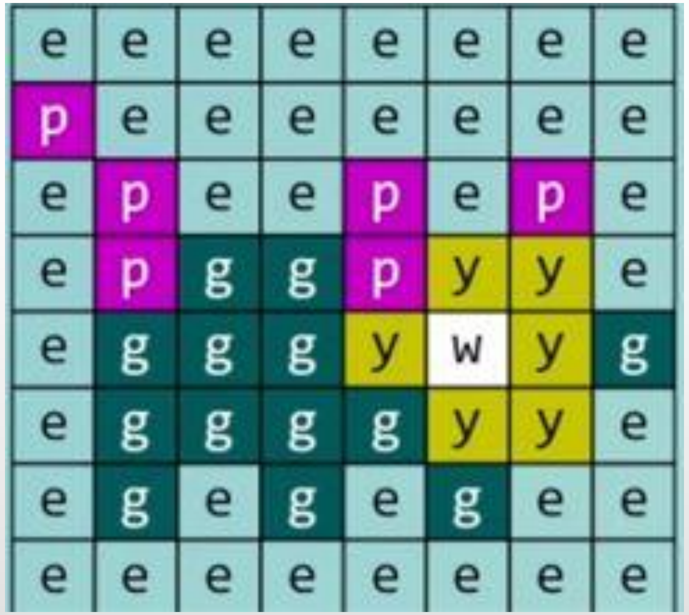
Translate Color Into Python Code



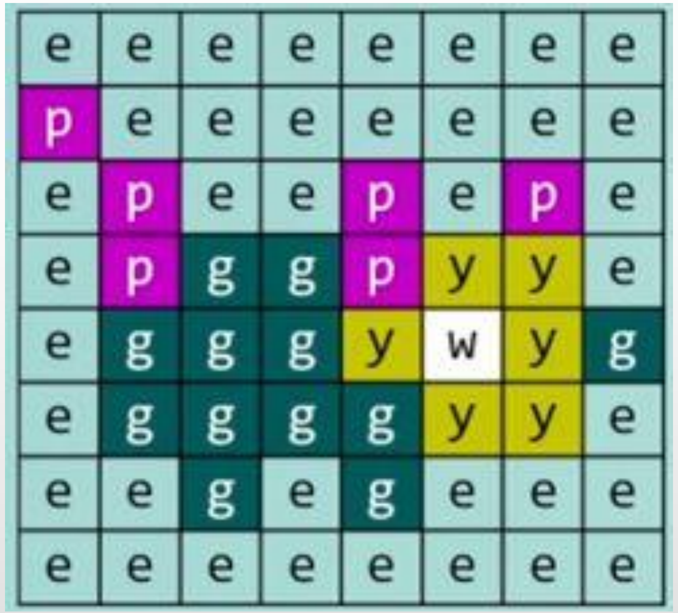
```
from sense_hat import SenseHat
import time

sense = SenseHat()

p=(204,0,204) #Pink
g=(0,102,102) #Green
w=(200,200,200) #White
y=(204,204,0) #Yellow
e=(0,0,0) #Empty
```

Label

- Label every single cell in your grid with the color



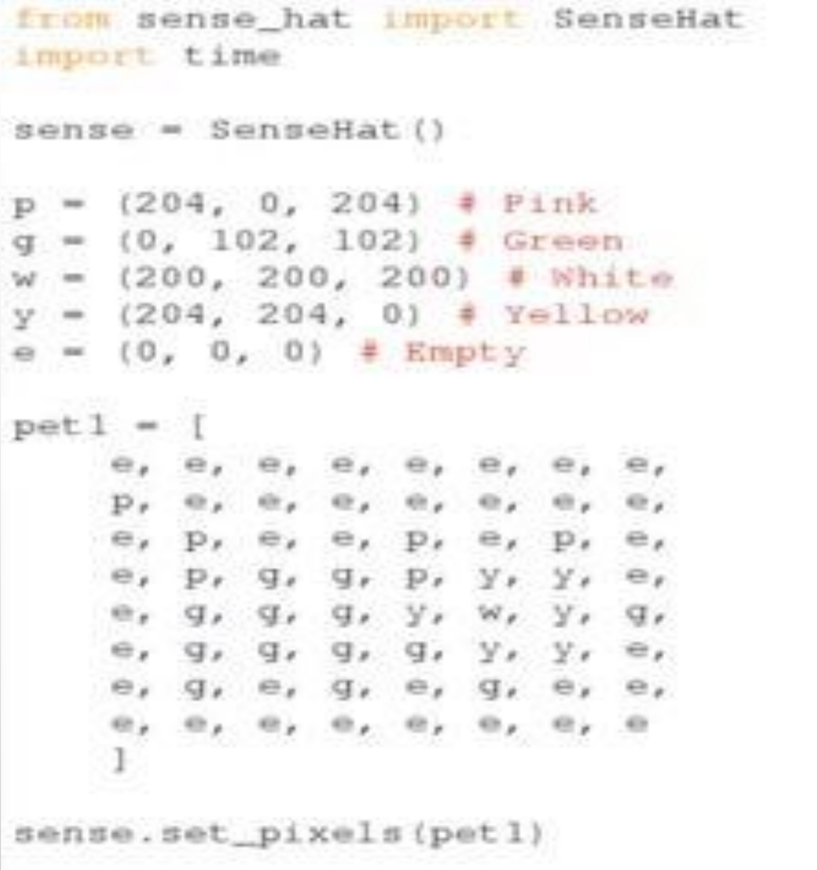
e	e	e	e	e	e	e	e
p	e	e	e	e	e	e	e
e	p	e	e	p	e	p	e
e	p	g	g	p	y	y	e
e	g	g	g	y	w	y	g
e	g	g	g	g	y	y	e
e	g	e	g	e	g	e	e
e	e	e	e	e	e	e	e



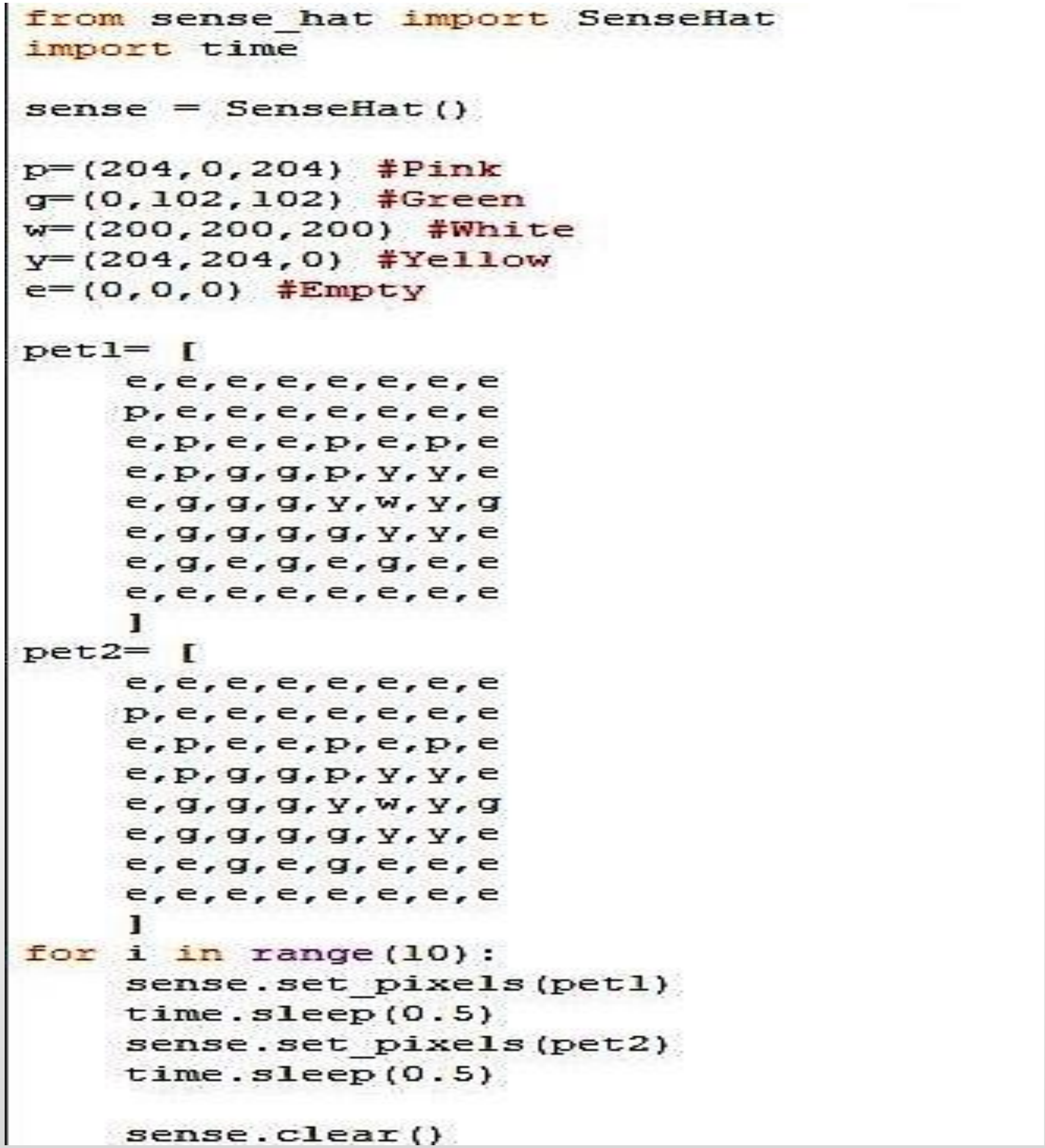
e	e	e	e	e	e	e	e
p	e	e	e	e	e	e	e
e	p	e	e	p	e	p	e
e	p	g	g	p	y	y	e
e	g	g	g	y	w	y	g
e	g	g	g	g	y	y	e
e	e	g	e	g	e	e	e
e	e	e	e	e	e	e	e

Translate Your Grid Into Python Code

- Using list and set_pixels to display your pet



```
from sense_hat import SenseHat
import time

sense = SenseHat()

p = (204, 0, 204) # Pink
g = (0, 102, 102) # Green
w = (200, 200, 200) # White
y = (204, 204, 0) # Yellow
e = (0, 0, 0) # Empty

pet1 = [
    e, e, e, e, e, e, e, e,
    p, e, e, e, e, e, e, e,
    e, p, e, e, p, e, p, e,
    e, p, g, g, p, y, y, e,
    e, g, g, g, y, w, y, g,
    e, g, g, g, g, y, y, e,
    e, g, e, g, e, g, e, e,
    e, e, e, e, e, e, e, e
    ]

sense.set_pixels(pet1)
```



```
from sense_hat import SenseHat
import time

sense = SenseHat()

p=(204,0,204) #Pink
g=(0,102,102) #Green
w=(200,200,200) #White
y=(204,204,0) #Yellow
e=(0,0,0) #Empty

pet1= [
    e,e,e,e,e,e,e,e
    p,e,e,e,e,e,e,e
    e,p,e,e,p,e,p,e
    e,p,g,g,p,y,y,e
    e,g,g,g,y,w,y,g
    e,g,g,g,g,y,y,e
    e,g,e,g,e,g,e,e
    e,e,e,e,e,e,e,e
    ]
pet2= [
    e,e,e,e,e,e,e,e
    p,e,e,e,e,e,e,e
    e,p,e,e,p,e,p,e
    e,p,g,g,p,y,y,e
    e,g,g,g,y,w,y,g
    e,g,g,g,g,y,y,e
    e,e,g,e,g,e,e,e
    e,e,e,e,e,e,e,e
    ]
for i in range(10):
    sense.set_pixels(pet1)
    time.sleep(0.5)
    sense.set_pixels(pet2)
    time.sleep(0.5)

    sense.clear()
```