



with the **Ursina 3D Engine**



Python Coding

with Ursina

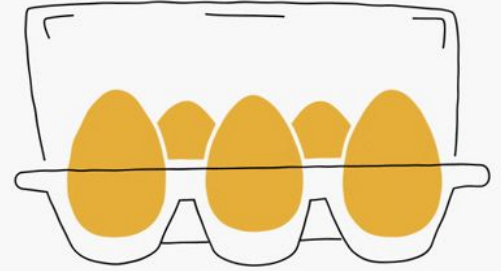
Course Stage: **Hello World!**

You will learn: -

- i) What **python** is, what ‘**modules**’ are, and whether python is **important** in the computing world today
- ii) How to **set up** a **project folder** for your python code with **Ursina**
- iii) How to code moving **3D objects** with **textures**, how to **set colours**, and to **print text** on screen

What is **python**?

- 1) Who invented it?
- 2) Is it popular?
- 3) Any examples of where it has been used?



What is **python**?

1) Guido van Rossum

2) Yes, very!

3) Google, Instagram, Netflix,
Data Science, Machine Learning...



Python uses modules

```
=====
>>> import random
>>>
>>> print(random.randint(1,10))
2
```

```
from ursina import *

app = Ursina()

app.run()
```

You can import **modules** at the top of your python code.

Different modules allow you to use **different functions**.

The **time** module (time.py) allows you to find the date, time events to the millisecond, and, of course, to get the time of day.

The **random** module (random.py) allows you to get random numbers.

The **ursina** module (ursina.py) allows you to...?

ursina engine



open source game engine

hello world!

Suggested steps:

- 1) Set up a window and get Ursina running
- 2) Create a 3D sphere
- 3) Put a texture on the sphere
- 4) Keep updating the sphere's rotation
- 5) Make a dark background, like space?

Let's make a **3D, rotating world** :)

So, what will we **basically** need to do to make our 3D (hello!) world?

Discuss ideas with a partner, then let's see how well you've **decomposed** the steps in this programming goal.



project setup and assets

1 Create a **new folder** on your H: drive called 'ursina projects'

2 Open **IDLE**

3 Create a new file called **hello.py** and save it in your new folder

4 Find a **planet texture**; also save it into your new ursina folder

Hello World!




```
from ursina import *
```

```
app = Ursina()
```

```
planet = Entity(model='sphere', texture='earth')  
planet.scale = 4
```

```
def update():
```

```
    planet.rotation_y += 0.4
```

```
app.run()
```

Here is our first **python** code. Just 7 lines :)

You must **save your .py file** before you can run it.

Make sure your **planet texture**, in your folder, is named **'earth'**.

Code Challenges!

- 1 Can you **change the rotation speed** of your planet?
- 2 Can you make your planet **change scale** (get bigger or smaller) **as it updates**?
- 3 Can you **add a small moon** -- with its own **texture**?



```
from ursina import *

app = Ursina()

planet = Entity(model='sphere', texture='earth')
planet.scale = 4

def update():
    planet.rotation_y += 0.4

app.run()
```

Changing the window's background colour

`window.color = color.black`

or

`window.color = color.rgb(0,0,0)`

or

`window.color = color.rgb(255,0,255)`

(shhh because pink)



```
message = Text('<bold>hello world!',background=True)
message.scale=2
message.background.color=color.lime
message.appear(speed=0.1)
```



We can **print text to the screen** :)

First, print '**hello world!**' to the screen.

Next, print whatever message you like.

Can you change the appearance of your text and its background? Can you make a transparent background?

Python Coding

with Ursina

Course Stage: **Hello World! 2.0**

You will learn: -

i) How to use **lists
and **iteration** to
create lots of
objects easily**

Code 1

```
p = Entity(model='cube',texture='earth')  
p = Entity(model='cube',texture='earth')  
p = Entity(model='cube',texture='earth')  
p = Entity(model='cube',texture='earth')  
p = Entity(model='cube',texture='earth')  
p = Entity(model='cube',texture='earth')  
p = Entity(model='cube',texture='earth')  
p = Entity(model='cube',texture='earth')  
p = Entity(model='cube',texture='earth')  
p = Entity(model='cube',texture='earth')  
p = Entity(model='cube',texture='earth')  
p = Entity(model='cube',texture='earth')  
p = Entity(model='cube',texture='earth')
```

Code 2

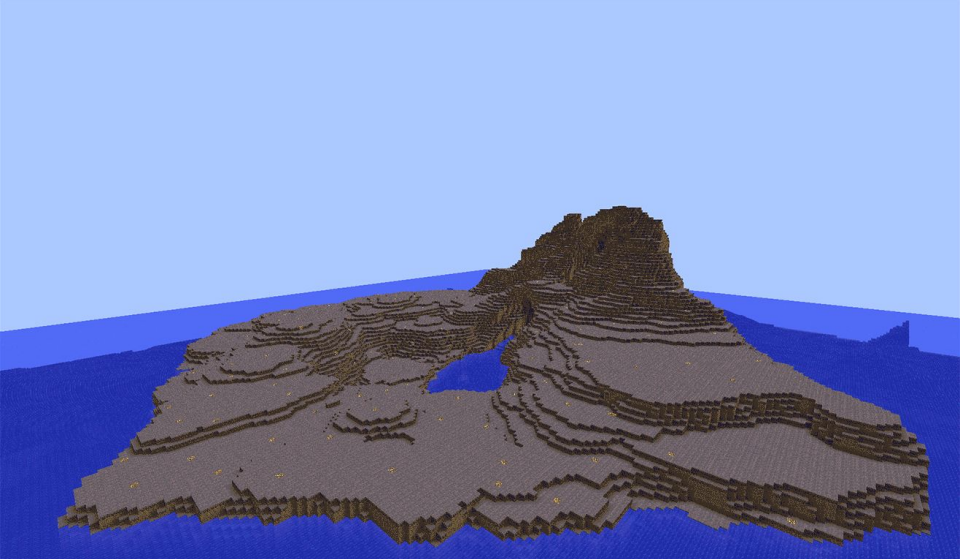
```
For i in range(28):  
    p = Entity( model='cube',  
                texture='earth')
```

Which bit of code,
code 1 or **code 2**,
will create the most
cubes?

Code 2

We can use a **for loop** to create hundreds...or *thousands* of objects :)

```
4  from ursina import *
5  from randVec3 import randVec3
6
7  app = Ursina()
8  window.color=color.pink
9  tex='earth' # Load our texture.
10 mod='cube' # Load the default cube model.
11
12 for i in range(280):
13     p = Entity(model=mod, texture=tex,
14               position=randVec3())
15
16 def update():
17     eye.rotation_y+=2*time.dt
18
19 eye = EditorCamera()
20 app.run()
```

Python Coding

with Ursina

Course Stage: **PythonCraft!**

You will learn: -

- i) What **Perlin noise** is and how to use it to code **Minecraft-like terrains**
- ii) How to **fix code** to get a **character moving** across your Minecraft world
- iii) How to use **lists** and **iteration** to add trees, mobs, and anything else



Download and place the **PythonCraft project folder** on your H: drive :)

Open the **pythonCraft.py** main module file :D

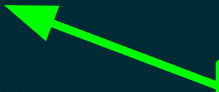
Extension

Can you **fix the code**? The **character** doesn't seem to be able to **move**.

What might be the **bug**?

```
17 # Initialise and set up our first-person character.
18 steve = Character(speed=0.01)
19
20 # Our main program update loop.
21 def update():
22     # Allow character to move over terrain.
23     steve.move(cambridge)
24
25 # Function that responds to key and mouse presses.
26 def input(key):
27     steve.input(key) # Character responds to 'escape' key.
28
29 # Start the program :)
30 app.run()
```

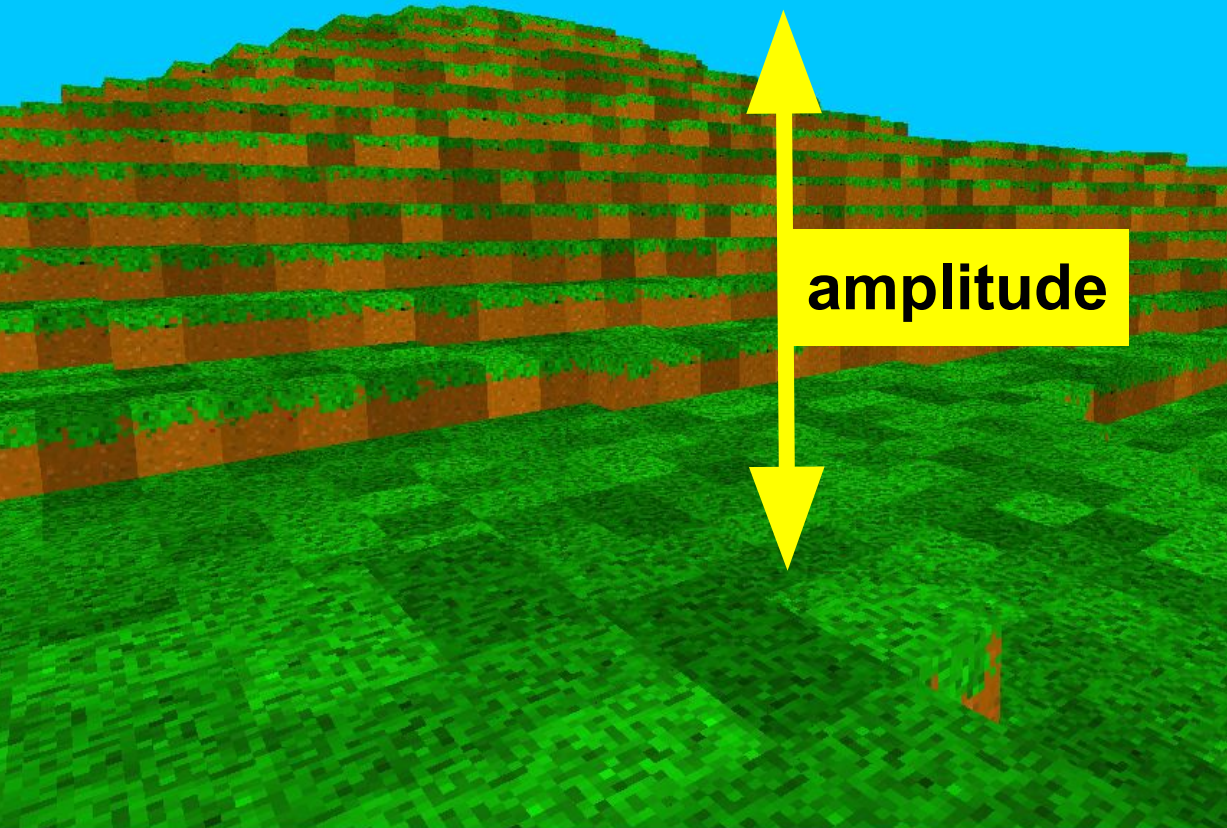
```
17 # Initialise and set up our first-person character.
18 steve = Character(speed=0.01)
19
20 # Our main program update loop.
21 def update():
22     # Allow character to move over terrain.
23     steve.move(cambridge)
24
25 # Function that responds to key and mouse presses.
26 def input(key):
27     steve.input(key) # Character responds to 'escape' key.
28
29 # Start the program :)
30 app.run()
```



This value is too small

```
# Initialise our terrain.
```

```
cambridge = Terrain(frequency=48, amplitude=32)
```



We pass in **two** arguments when making a **Terrain()** object.

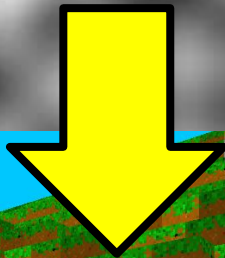
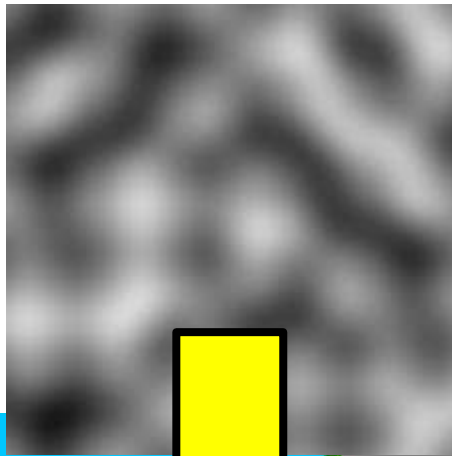
It uses a '**Perlin noise**' function to generate the hills and valleys of our terrain.

The **amplitude** value is how **high** or **low** the terrain can get.

frequency?



Prof. Ken
Perlin



6 -11 53



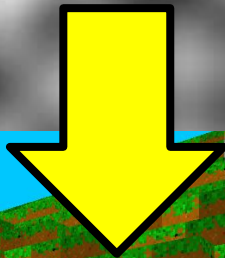
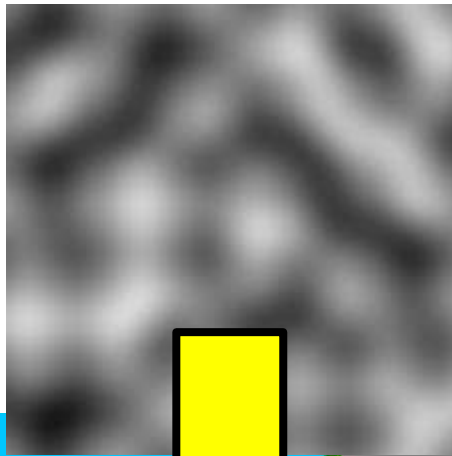
Generating Terrain with **Perlin Noise**

amplitude - how high/low

frequency - the smoothness
of amplitude variation from
one area of the terrain to the
next



Prof. Ken
Perlin



6 -11 53

Generating Terrain with Perlin Noise

```
cambridge = Terrain(advanced=True,  
                    a1=100, f1=124,  
                    a2=50, f2=80,  
                    a3=1, f3=1)
```

octaves

We can make more realistic and interesting terrains by passing in **three amplitudes** with **three frequencies**.

