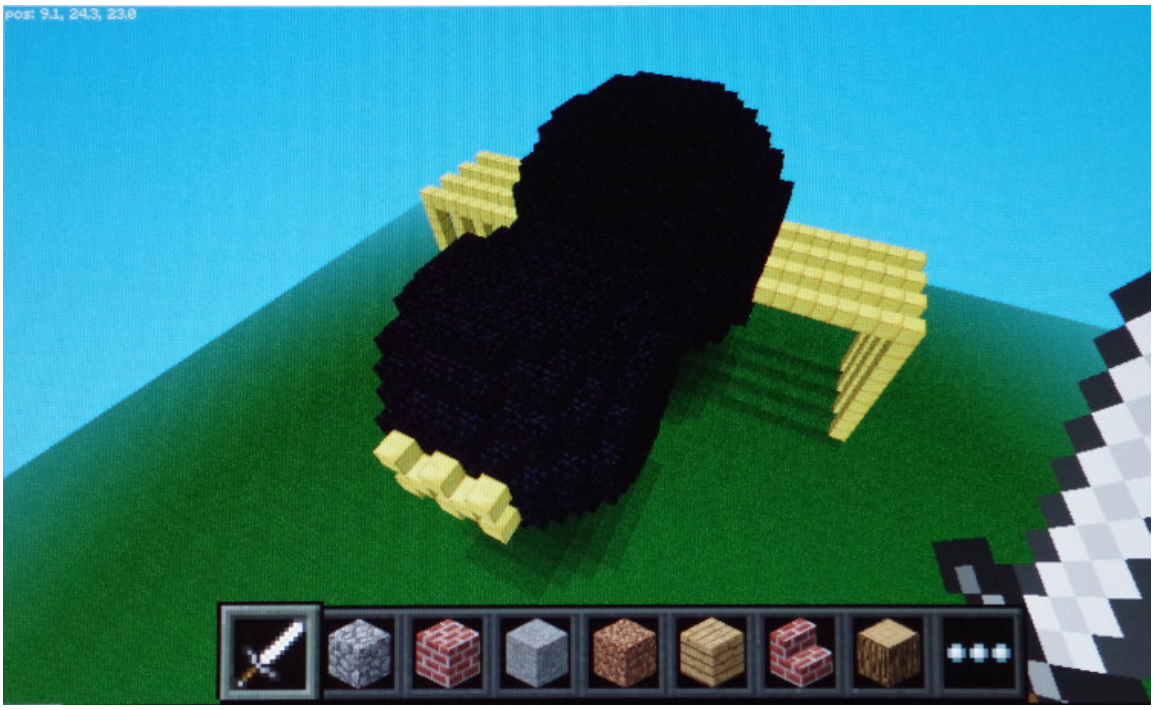
A Minecraft spider generator



Spider

Before you start:

- Launch Minecraft and create a new world.
- Launch Geany

Step 1 Make a body

- In Geany, from the file menu select "new", and then save as `spider.py` in the "pi" directory.
- We need to import the dBsCode commands we'll be using and clear an area in the Minecraft world for working in. Create this program:

```
from dbscode_minecraft import *
bulldoze()
```

- **Test** Press F5 to run the program (this will also save your program for you). After a few seconds you should see a "flatworld" type of environment.
- Our spider will consist of a body, a head and some legs.
- Lets start with the body. Build a sphere by adding this to the end of your script:

```
sphere(DIAMOND_BLOCK, point(0, 10, 0), 8)
```

- **Test** Press F5 and navigate to the centre of the world (using the coordinates at the top left of the Minecraft window). You should see a sphere shape. This shape has it's centre at 0, 10, 0 and a radius of 8. The diameter of the sphere is actually 15 blocks 2 smaller than 2 * radius, remembering this maybe helpful later on.
- Not very exciting yet, lets add a head, we make it a bit smaller than the body:

```
sphere(DIAMOND_BLOCK, point(0, 10, 8 + 6), 6)
```

- **Test** Press F5 and make sure there are 2 spheres drawn. We've positioned the smaller one in front of the large sphere by moving it's centre by the radius of each sphere. Can you see the gap? Remember the diameter of a sphere is 2 blocks less than the radius. See if you can fix it. The correct code is shown further on.

Step 2 Make a leg

- Okay, we need to draw some legs, we could draw a line straight down, but let's give the spider some knees too. We will draw a horizontal line out from the body at point(8, 10, 0), this is 8 blocks long and one thick, point(8, 1, 1)

```
box(OBSIDIAN, point(8, 10, 0), point(8, 1, 1))
```

- Then we draw a vertical line down. We position this underneath the last horizontal block at point(15, 9, 0) and draw 7 blocks down using point(1, -7, 1)

```
box(OBSIDIAN, point(15, 9, 0), point(1, -7, 1))
```

Step 3 Make a function for drawing legs

- But you need to draw 8 legs, 4 on each side. This is where we can use something very useful in programming called a 'function'. Functions let you write some code that you can use over and over again by 'calling' the function repeatedly.
- You are going to write a function that draws the 2 boxes above.

- We use 'def' to build functions. Collect together the 2 lines beginning with "box" that you've just written so it looks like this
- the spaces at the start of the lines are important - there are 4 of them, or just press the "tab" key once:

```
def leg():
    box(OBSIDIAN, point(8, 10, 0), point(8, 1, 1))
    box(OBSIDIAN, point(15, 9, 0), point(1, -7, 1))
```

Now we can "call" this function by simply adding this to the bottom of your program. Your code should now look something like this:

```
sphere(DIAMOND_BLOCK, point(0, 10, 0), 8)
sphere(DIAMOND_BLOCK, point(0, 10, 7 + 6), 6)

def leg():
    box(OBSIDIAN, point(8, 10, 0), point(8, 1, 1))
    box(OBSIDIAN, point(15, 9, 0), point(1, -7, 1))

leg()
```

- Press F5 - your spider should appear with one leg, now we need to make it draw the other 7.
- We can add a position "parameter" to the function. We use parameters to pass information into a function by adding them to the brackets at the top and refering to them inside. Change your existing function and add "pos" to all of the positions of the shapes:

```
def leg(pos):
    box(OBSIDIAN, pos, point(8, 1, 1))
    box(OBSIDIAN, pos+point(7, -1, 0), point(1, -7, 1))
```

- We can now pass in the position when we call the function to draw a leg in any position. This will draw 4 legs on one side of the body, spaced apart by a block. Change the "leg()" line to the following:

```
leg(point(8, 10, 1))
leg(point(8, 10, 3))
leg(point(8, 10, 5))
leg(point(8, 10, 7))
```

- Press F5 - your spider should appear with 4 legs on one side, but they may be slightly out of position. See if you can fix this. (Remember the middle of the sphere is at "0")

Step 4 Loop the loop.

- Programmers are lazy... We don't want to write the same thing over and over, so we made functions, but we also made loops which allow us to do the same thing over and over again, maybe changing one small thing. See above where we make 4 legs - the only thing that changes is the last number (the "z" position) of the leg.
- To do this we need a "for" loop. Remove the 4 leg() "calls" and replace with this:

```
for i in range(0, 4):
    leg(point(8, 10, i * 2))
```

- This will repeat the `leg` line 4 times (make sure you add 4 spaces before), each time with i being a number described by the `range` function - so 0 to 4 in this case. We set the z position of the leg by multiplying this by 2 (so each house in the row appears spaced by 2 blocks)
- But the position is still wrong. We can change our range above so that it positions the legs with negative and positive z positions, change the above for loop to do this:

```
for i in range(-2, 2):
    leg(point(8, 10, i * 2 + 1))
```

- Press F5 to test again - your legs should now be positioned either side of the vertical centre line of the sphere.

Step 5 Repeat after me.

So now we need to draw some legs the other side of the spider, we need to do two things:

- Position the legs on the other side of the body - this means changing one of the parameters for our point to a negative value. We can be lazy and use the same loop again, bonus! Copy the code which calls the leg function and repeat it underneath the line you copied:

```
for i in range(-2, 2):
    leg(point(8, 10, i * 2 + 1))
    leg(point(8, 10, i * 2 + 1))
```

- Right, now change one of the three parameters to a negative value. See if you can work out the correct one to do before you press F5.
- This code will only move the legs to start at the other side of the spider, they will

disappear inside it if you are correct. If you get it wrong, try again. The correct code will be shown later.

Step 6 Flip it.

- The second thing we need to do is turn the legs around so they stick out from the spider. We can use our function to help us again and pass in a Boolean (True or False) parameter which says whether to flip them around the other way.
- Because of the way the sphere is drawn, we need to jiggle our maths a little to make the legs draw correctly. The code below shows the function and the for loop which calls it - replace your leg drawing code with this:

```
def leg(pos, flip):
    if(flip):
        box(GOLD_BLOCK, pos+point(-7, 0, 0), point(8, 1, 1))
        box(OBSIDIAN, pos+point(-7, -1, 0), point(1, -7, 1))
    else:
        box(GOLD_BLOCK, pos, point(8, 1, 1))
        box(OBSIDIAN, pos+point(7, -1, 0), point(1, -7, 1))

for i in range(-2, 2):
    leg(point(8, 10, i * 2 + 1), False)
    leg(point(-8, 10, i * 2 + 1), True)
```

Step 7 Eyes.

- Spiders have 8 eyes, so we will try and give our creature some. We will do this programmatically, but remember how the sphere had an odd diameter? This means we'll have to be arrange our eyes so they don't look off centre. Computers are good at loops so we will use this to make a check pattern.
- We also want the eyes to be at the front of the head, so we need to move them from the centre of the spider forward - how far do you think this is?
- The distance is the radius of the body + the diameter of the head.
- First we will write some code to draw 8 blocks in a 3 x 2 check pattern. We will make 2 loops to do 3 rows of 5 columns and draw a block every other loop. We use a Boolean "flag" like the flip for the legs to decide whether to draw a block. Each time we go through our loop we "toggle" the flag, to be the opposite of what it was last time. This means we can do a check pattern quite simply.
- Add the following code to the bottom of your file:

```
draw = True
for x in range(-2, 3):
    for y in range(-1, 2):
        if True == draw:
            box(GOLD_BLOCK, point(x, y + 10, 20), point(1, 1, 1))
        draw = ~draw
```

- Press F5 to test and you should see the eyes appear in a check pattern in front of the spiders head. We have a gap between the head and eyes because we chose "20" for the z- position just to get it working.

- Right, lets put this in a function so we can tell it where to put the eyes easily. Remove the code you just wrote and replace it with the following:

```
def eyes(pos):
    draw = True
    for x in range(-2, 3):
        for y in range(-1, 2):
            if True == draw:
                box(GOLD_BLOCK, point(pos.x + x, pos.y + y, pos.z),
            draw = ~draw

eyes(point(0, 10, 19))
```

Step 8 Tidy up.

- Right we've got a spider like thing on the screen, but what about those colours?

- Lets make a function to draw the spider with a colour for the body and head and another colour for the legs and eyes.

- Whilst we are here, let's make it so you can draw the spider in any position on the map.

- We need to move our functions round a little, so some copy and paste might be useful. Try to move all your "def" functions above your function calls so your code looks like this:

```
from dbscode_minecraft import *

bulldoze()

def leg(pos, flip):
    if(flip):
        box(GOLD_BLOCK, pos+point(-7, 0, 0), point(8, 1, 1))
        box(OBSIDIAN, pos+point(-7, -1, 0), point(1, -7, 1))
```

```
    else:
        box(GOLD_BLOCK, pos, point(8, 1, 1))
        box(OBSIDIAN, pos+point(7, -1, 0), point(1, -7, 1))

def eyes(pos):
    draw = True
    for x in range(-2, 3):
        for y in range(-1, 2):
            if True == draw:
                box(GOLD_BLOCK, point(pos.x + x, pos.y + y, pos.z),
            draw = ~draw

sphere(DIAMOND_BLOCK, point(0, 10, 0), 8)
sphere(DIAMOND_BLOCK, point(0, 10, 7 + 6), 6)
for i in range(-2, 2):
    leg(point(8, 10, i * 2 + 1), False)
    leg(point(-8, 10, i * 2 + 1), True)

eyes(point(0, 10, 19))
```

- Now create a spider function which draws the 2 spheres, the legs and eyes. This needs to take 3 parameters, the colours and the position.

```
def spider(body_colour, leg_colour, pos):
```

- Which code goes inside the function?
- How would you call the function?
- Can you change the sphere functions to use the body colour?
- Can you change the sphere functions to draw the body and head in the position you chose?
- Can you add another parameter to the leg and eye functions so you can tell it the colour you want?
- Can you reposition the legs with the position of the spider?
- Here is the whole code that should do all this:

```
from dbscode_minecraft import *

bulldoze()

def leg(pos, flip, colour):
    if(flip):
        box(colour, pos+point(-7, 0, 0), point(8, 1, 1))
```

```
        box(colour, pos+point(-7, -1, 0), point(1, -7, 1))
    else:
        box(colour, pos, point(8, 1, 1))
        box(colour, pos+point(7, -1, 0), point(1, -7, 1))

def eyes(pos, colour):
    draw = True
    for x in range(-2, 3):
        for y in range(-1, 2):
            if True == draw:
                box(colour, point(pos.x + x, pos.y + y, pos.z), poin
            draw = ~draw

def spider(body_colour, leg_colour, pos):
    sphere(body_colour, pos, 8)
    sphere(body_colour, pos+point(0, 0, 13), 6)
    for i in range(-2, 2):
        leg(pos+point(8, 0, i * 2 + 1), False, leg_colour)
        leg(pos+point(-8, 0, i * 2 + 1), True, leg_colour)

    eyes(pos+point(0, 0, 19), leg_colour)

spider(OBSIDIAN, GOLD_BLOCK, point(0, 10, 0))
```

Step 9 Spidercraft

- So you can make a spider, but can you make more of them? How might you do that?
- Could you make 3 spiders in a line with different colours?
- How might you make things a bit more random, could you make each spider have a random size? If you do this, how do you make the head and legs stay in proportion to the body?
- What happens if you try to draw 10 spiders?
- Could you invent some new creatures?
- Could you re-use some of the code to draw a snowman?