Python coding for Minecraft

by **arpruss** on March 28, 2015

Table of Contents

Intro: Python coding for Minecraft

This Instructable shows how to install and use a mod I wrote that lets you control Minecraft with python scripts. I'll focus on Windows, though OS X and Linux should work just as well.

Python scripts can generate neat in-world things. With a few lines you can draw a giant glass sphere, or with a bit more work make a giant Sierpinski triangle in the sky. I've made fun scripts for a water-filled glass donut, a gigantic Klein bottle and to turn everything around into TNT. There is a whole book introducing programming using python scripts for Minecraft, and you can even make simple Minecraft-based games.

For a while now you could write python scripts for Minecraft on the Raspberry Pi. I wanted my kids to be able to do that, but we don't have a Pi, plus it would be nice to do this with the full desktop Minecraft. You could run your own server with the Raspberry Juice plugin which enables most of the python scripts to work. But not everyone wants to install and configure a server.

So I wrote the Raspberry Jam mod for Minecraft 1.8 that emulates most of the Raspberry Pi Minecraft protocol (about the same as the Raspberry Juice plugin provides) and lets Raspberry Pi python scripts run with full desktop Minecraft.

I later found out that someone wrote the mcpiapi mod for Minecraft 1.7.10 a couple of weeks earlier. This Instructable is written for Raspberry Jam mod and Minecraft 1.8, but I'll note the differences for mcpiapi and Minecraft 1.7.10. Most of my work was with Python 2.7, but I'll also note what to do if you prefer Python 3.

I assume that you have basic facility with downloading, unzipping, creating folders and copying files on Windows (or your operating system of choice).

Image Notes
1. generated with donut.py

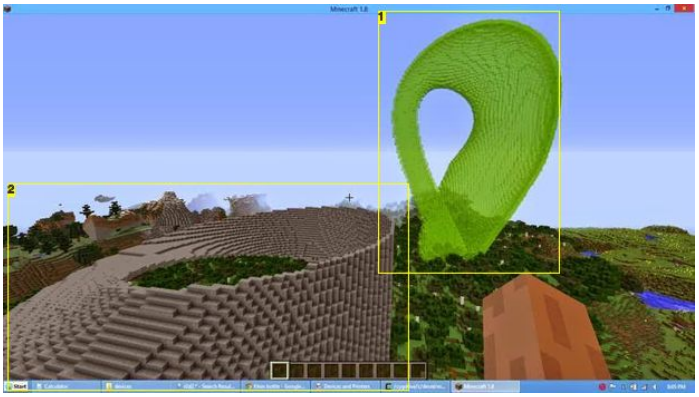
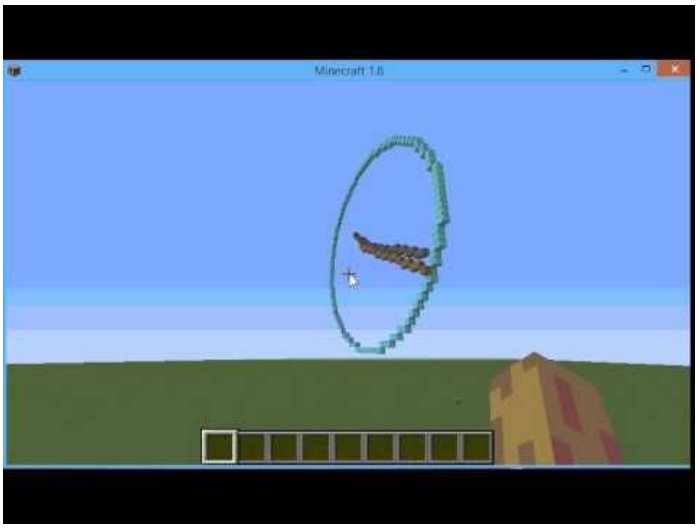


Image Notes
1. from klein.py
2. from mobius.py

Image Notes
1. real-time working analog clock from stuffaboutcode.com, in stuffaboutcode_clock.py



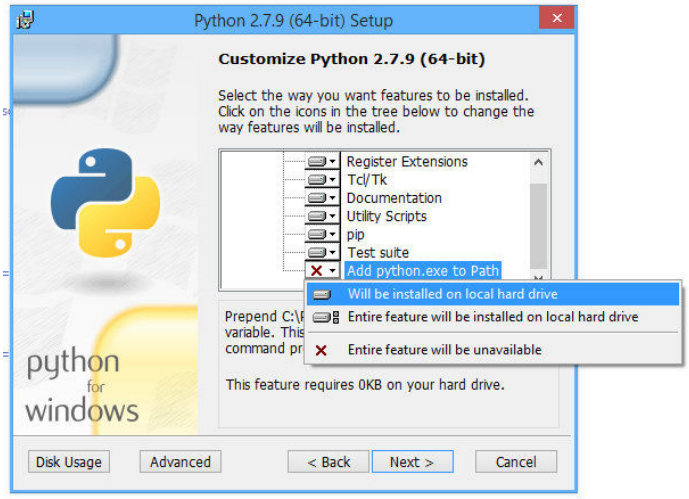
Step 1: Install Python

You need to decide if you want Python 2.7 or Python 3.x. The Adventures in Minecraft book uses 2.7, and most of the scripts floating around the web are for 2.7, but I have converted a lot of scripts for 3.x if you prefer.

1. Download your choice of Python installed from here.

2. Run the installer.

3. Click through to the Customize Python dialog, and make sure to scroll down to "Add python.exe to path", click on it and select "Will be installed on local hard drive." If you don't add Python to the path, you won't be able to launch scripts with **/python** from within Minecraft.



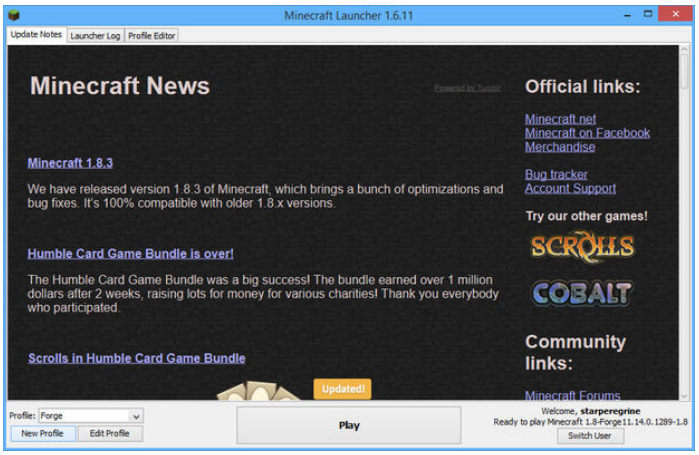
Step 2: Install Forge for Minecraft

The Forge manages Minecraft mods, and is needed for the Raspberry Jam Mod (and the mcpapi mod for 1.7.10 as well).

I assume you have Minecraft installed.

1. You need to run Minecraft 1.8 once (and it needs to be exactly 1.8, not 1.8.3, say). To do that, start the Minecraft Launcher, and after logging in, click on New Profile. Set the profile name to anything you want (e.g., "Test of 1.8") and then go to "Use version" and select "Release 1.8". Then click on "Save Profile", make sure the new profile is selected in the launcher, and click on "Play". Start a world and make sure it works.
2. Exit Minecraft and Minecraft Launcher.
3. Download Forge for 1.8 installer. The version that I've been using is 11.14.0.1289 and can be directly downloaded here.
4. Run the Forge installer. Default settings should work.
5. Start Minecraft. You will now have a new Forge profile.

1.7.10 variant: Make a 1.7.10 profile in step 1, get your 1.7.10 Forge from here in step 3.



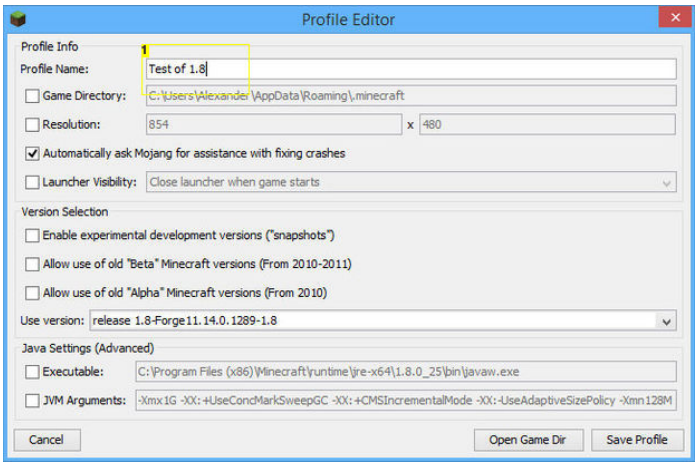
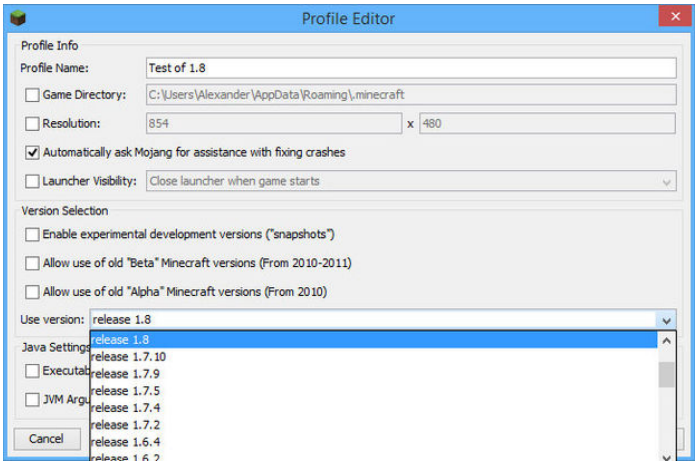


Image Notes
1. Make a name for test profile



Step 3: Install mod and scripts

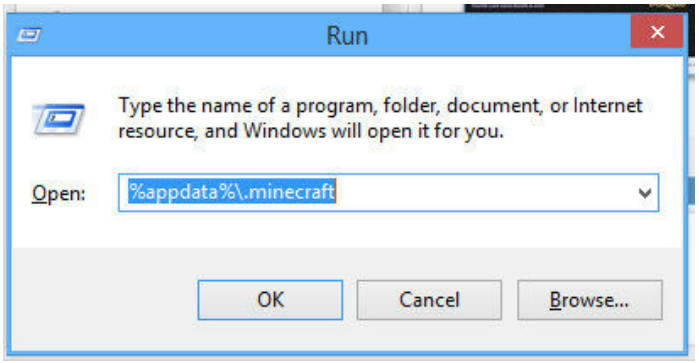
1. Create a **mods** folder in your Minecraft folder. (To do that, press **Windows-R**, type **% appdata%\.minecraft**, press **enter**. You will see everything in your Minecraft folder. If you already have a **mods** subfolder, you don't need to do anything. Otherwise, make one. On Windows 8, click on **New folder**, and then type in the name **mods** and press **enter**.)

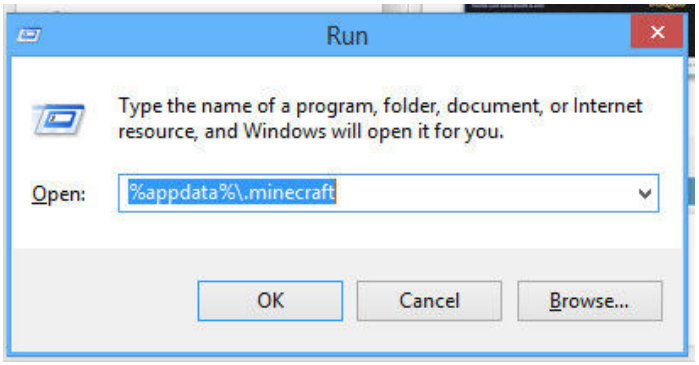
2. Download the .jar file from the latest version of the Raspberry Jam Mod to the **mods** folder you just made (or found). For instance, you can just right click on the jar file (e.g., **RaspberryJam-Mod-0.08.jar**) in your browser, choose **"Save link as..."** and then type **%appdata%\.minecraft\mods\RaspberryJam-Mod-0.08.jar** (or whatever version you're downloading) under "filename".

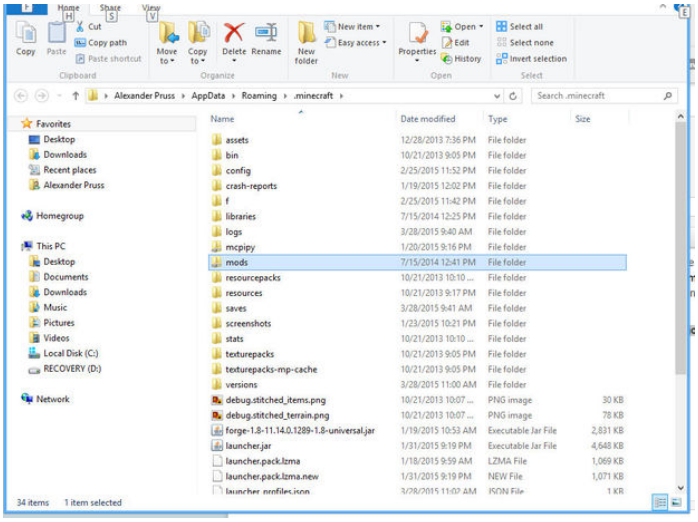
3. Download the zip file containing sample Python scripts and the mcpi library. The Python 2 version is here and the Python 3 version is here.

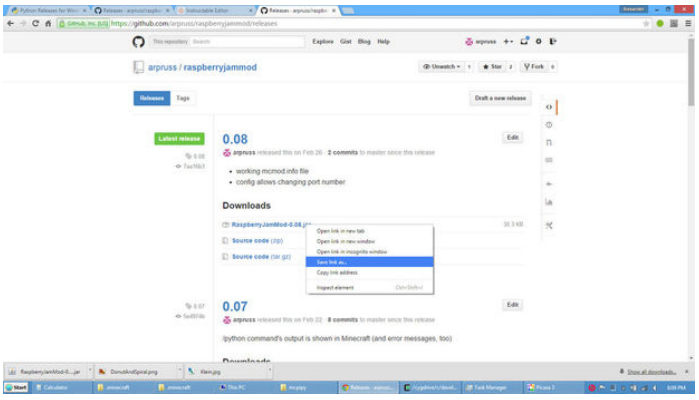
4. Open the downloaded zip file (in Chrome, by clicking on it at the bottom of the window). It has a **mcpipy** folder. Copy the **mcpipy** folder into your Minecraft folder. (To do that, click once on the **mcpipy** folder in the zip file, and press **ctrl-c**, then navigate to the **%appdata%\.minecraft** folder as in step 1, and press **ctrl-v**).

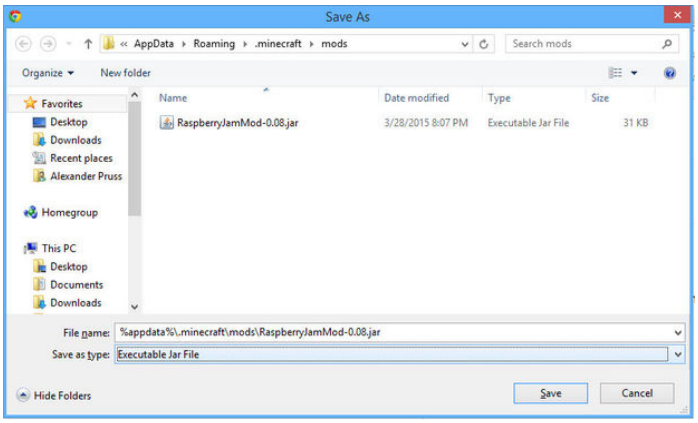
Variant for 1.7.10: In Step 2, download the .jar file from here. (After clicking in the .jar file in the browser, click on **Raw** to download.) In Step 4, you will need to make a **mcpimods** folder in your **%appdata%\.minecraft** folder, and a **python** folder in the **mcpimods** folder. Then put the contents of the mcpipy folder in the scripts zip file in the **%appdata%\.minecraft\mcpimods\python** folder.

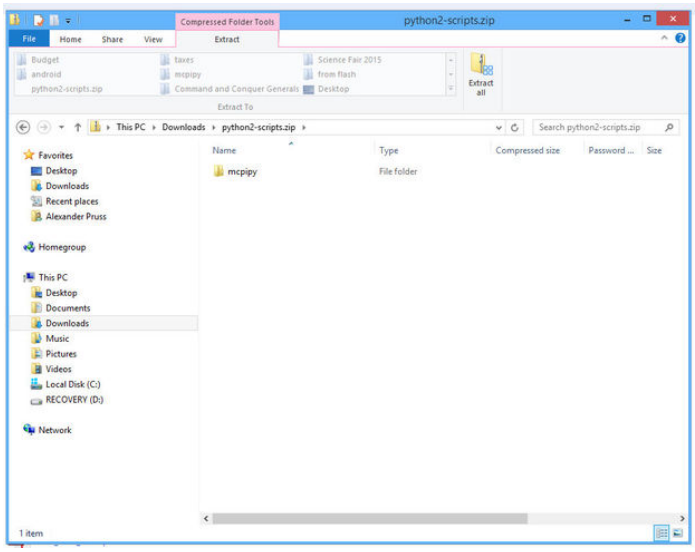



Run
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.
Open:
%appdata%\.minecraft
OK
Cancel
Browse...


Alexander Pruss ▸ AppData ▸ Roaming ▸ .minecraft ▸
Search .minecraft
Favorites
Desktop
Downloads
Recent places
Alexander Pruss
Homegroup
This PC
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
RECOVERY (D:)
Network
Name
Date modified
Type
Size
assets
bin
config
crash-reports
f
libraries
logs
mcpipy
mods
resourcepacks
resources
saves
screenshots
stats
texturepacks
texturepacks-mp-cache
versions
debug.stitched_items.png
debug.stitched_terrain.png
forge-1.8-11.14.0.1289-1.8-universal.jar
launcher.jar
launcher.pack.lzma
launcher.pack.lzma.new
34 items
1 item selected


arpruss / raspberryjammod
Releases
Tags
0.08
working mcmod.info file
config allows changing port number
Downloads
0.07
/python command's output is shown in Minecraft (and error messages, too)


Save As
AppData ▸ Roaming ▸ .minecraft ▸ mods
Search mods
Organize
New folder
Favorites
Desktop
Downloads
Recent places
Alexander Pruss
Homegroup
This PC
Desktop
Documents
Downloads
Name
Date modified
Type
Size
RaspberryJamMod-0.08.jar
3/28/2015 8:07 PM
Executable Jar File
31 KB
File name:
%appdata%\.minecraft\mods\RaspberryJamMod-0.08.jar
Save as type:
Executable Jar File
Hide Folders
Save
Cancel


Compressed Folder Tools
python2-scripts.zip
File
Home
Share
View
Extract
Extract all
Extract To
This PC ▸ Downloads ▸ python2-scripts.zip ▸
Search python2-scripts.zip
Favorites
Desktop
Downloads
Recent places
Alexander Pruss
Homegroup
This PC
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
RECOVERY (D:)
Network
Name
Type
Compressed size
Password ...
Size
mcpipy
File folder
1 item

Step 4: Testing mod

Start Minecraft, making sure that you use the **Forge** profile.

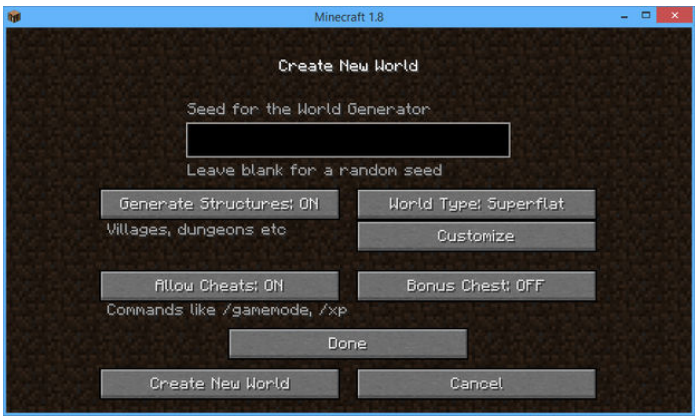
Create a new world (it's too easy to mess things up with the python scripts). My daughter suggested I use Creative and Superflat.

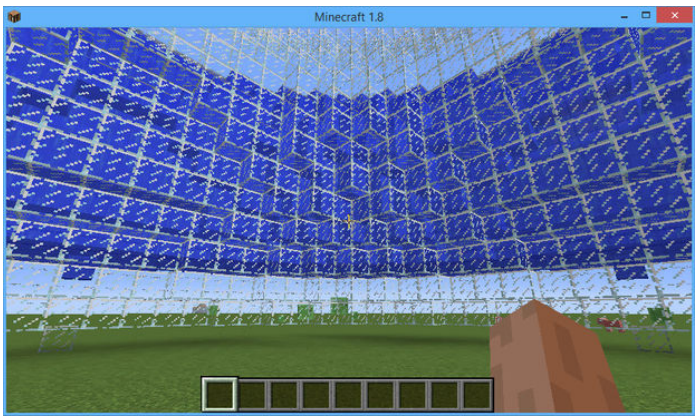
In Minecraft, type **/python donut** and press **enter**.

If all goes well, a giant glass donut will be drawn around you, and then it will be filled with water.

If you get something like a **Script not found** error, this probably means that you don't have the sample scripts installed in the **%appdata%\.minecraft\mcpipy** folder (or **%appdata%\.minecraft\mcpimods\python** for 1.7.10, but the error message may be different for 1.7.10).

If you get a **'Cannot run program "python"'** error, you don't have your python directory in your system path. You may want to add it manually to the path, or just reinstall python, following the directions in Step 3 of my python install step.





Step 5: Getting started programming Minecraft in python

The easiest way to get started programming Minecraft in python is to start with one of the simpler sample scripts. I recommend making a shortcut on your desktop to the scripts folder (**%appdata%\.minecraft\mcpipy** for 1.8 or **%appdata%\.minecraft\mcpimods\python** for 1.7.10).

In your scripts directory, you can right click on any script and you should have an **Edit with IDLE** option. A fun script to modify is my water-filled donut script (**donut.py**). For instance, change block.WATER to block.GRASS in the second last line to make a silly grass-filled donut. You can run this with **/python donut** from Minecraft, or just by pressing **F5** in IDLE.

Or to make a simple new script, create a **helloworld.py** file with your favorite text editor (even Notepad). Put at the top:

```
import mcpi.minecraft as minecraft
import mcpi.block as block
import server
import sys
from math import *
```

This imports the needed library code. Connect to Minecraft with:

```
mc = minecraft.Minecraft.create(server.address)
```

You can then send a "Hello world!" message to the user with:

```
mc.postToChat("Hello world!")
```

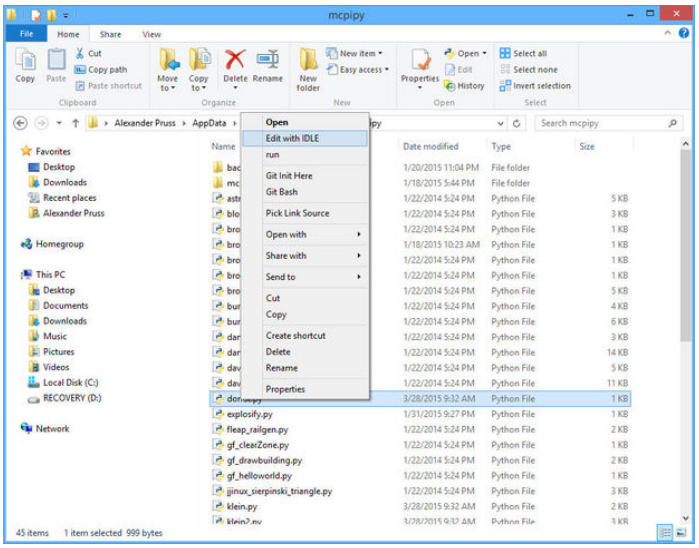
If you want to create a diamond block right under the player, you can also do:

```
playerPos = mc.player.getPos()
mc.setBlock(playerPos.x,playerPos.y-1,playerPos.z,block.DIAMOND_ORE)
```

The coordinates for setBlock() and getPos() are measured from the player's spawn point (which is thus (0,0,0)).

(For a list of all the block.* values, see **mcpi\block.py** in the scripts folder. You can also directly use Minecraft block numbers.)

To run your script, save it and type **/python helloworld** in your Minecraft world and press **enter**.



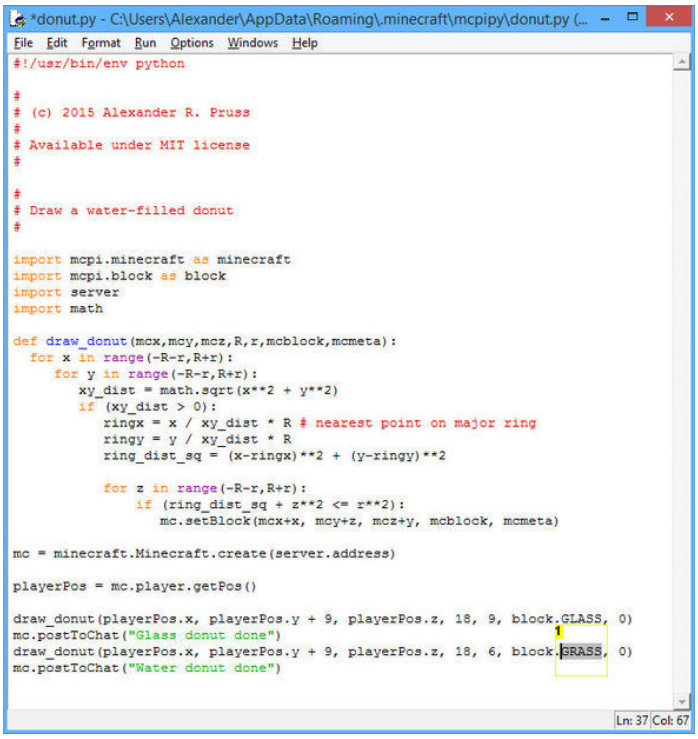
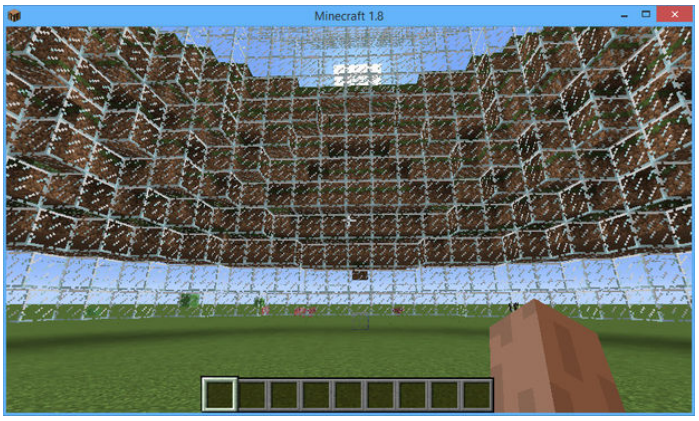
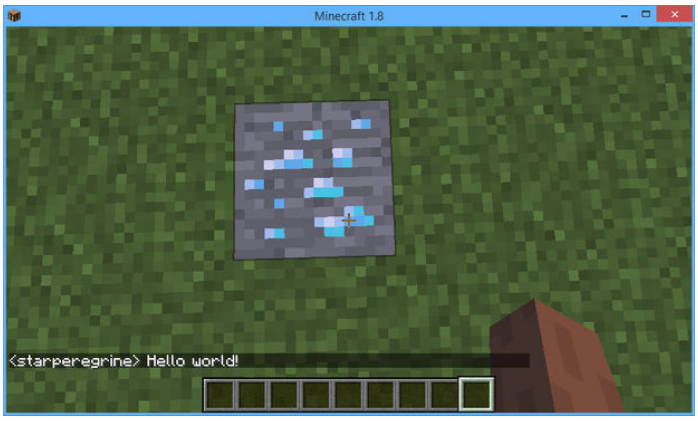


Image Notes
1. Used to be WATER.





Step 6: Advanced notes 1: Drawing objects defined by an inequality

There are basically two different techniques for drawing mathematically defined objects with a python script in Minecraft.

One way is to define a solid object by an inequality. For instance, a sphere centered on (x0,y0,z0) with radius r can be defined by the inequality:

```
(x-x0)**2 + (y-y0)**2 + (z-z0)**2 <= r**2
```

(I.e., the distance to (x0,y0,z0) is at most r.) So to draw a sphere, just loop through all points (x,y,z) in a cube of side-length 2*r+1 surrounding (x0,y0,z0), and draw a block if the above inequality holds.

I learned this technique from the sample **nt7s_sphere.py** script. Start with the standard header and init Minecraft connection code:

import mcpi.minecraft as minecraft
import mcpi.block as block import server

mc = minecraft.Minecraft.create(server.address)

Then just do:

radius = 8

playerPos = mc.player.getPos()

```
for x in range(radius*-1,radius):
 for y in range(radius*-1, radius):
   for z in range(radius*-1,radius):
     if x**2 + y**2 + z**2 < radius**2:
       mc.setBlock(playerPos.x + x, playerPos.y + y + radius, playerPos.z - z - 10, block.GLASS)
```

This draws a sphere of the specified radius above the player, and a little offset in the z-direction.

I use the same technique, but with a more complicated formula, in my **donut.py** script:

```
for x in range(-R-r,R+r):
    for y in range(-R-r,R+r):
       xy_dist = math.sqrt(x**2 + y**2)
       if (xy_dist > 0):
          ringx = x / xy_dist * R # nearest point on major ring
          ringy = y / xy_dist * R
          ring_dist_sq = (x-ringx)**2 + (y-ringy)**2
          for z in range(-R-r,R+r):
              if (ring_dist_sq + z**2 <= r**2):
                 mc.setBlock(mcx+x, mcy+z, mcz+y, mcblock, mcmeta)
```

While the inequality technique works best for solid shapes, you can use it for hollow shapes in two ways. One way is to use two inequalities, for instance in the case of the sphere one to make sure that we're within the outer radius of the center and another to make sure we're not closer than the inner radius. The other way is just to draw another object with smaller dimensions made out of air inside the larger solid object, much as in my **donut.py** script, I initially draw a glass donut, and then replace the inside of it with water.

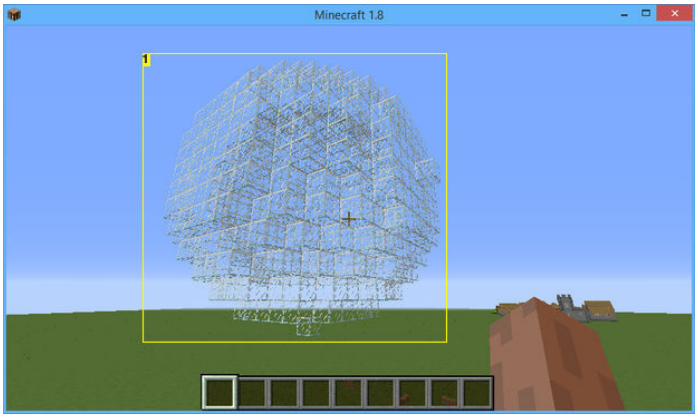


Image Notes
1. nt7s_sphere.py

Step 7: Advanced notes 2: Drawing an object defined by a parametric surface

One can also draw a surface by parametrizing it with two parameters, say a and b, and then looping over a range of these parameters, setting blocks where needed. In my Klein bottle and Mobius strip scripts, I have a general draw_surface() method that lets one do this. For instance, the Mobius strip (see my **mobius.py** script) is defined by the three formulae:

```
x = (3 + a * cos(b/2)) * cos(b)
y = a * sin(b/2)
z = (3 + a * cos(b/2)) * sin(b)
```

with a ranging from -1 to 1 and b ranging from 0 to 2*pi (yet another reason why this can be in the pi/e contest?). You can think of b as defining the angle around the circuit, and a moving one from one edge to the other.

Using scripts like this, you need to ensure that in your loops over a and b, the steps are sufficiently small that there are no gaps in the surface. Unless that's the effect you're after.

For details and examples see **mobius.py**, **klein.py** and **klein2.py**.

Step 8: Advanced notes 3: Knots

You can find parametric equation for knots on the net. This time, we're going to do things slightly different from before. Before, we had loops driving calls to mc.setBlock() directly. But in our surface plots, such as the Klein bottle, often the same block would get drawn multiple times, which is slow and inefficient. A better way is to assemble the geometric figure in memory, and then to write it all at once.

Let me go through an example like that (in **knot.py**) to draw a knot. Start with a standard header like:

```
import mcpi.minecraft as minecraft
import mcpi.block as block import server from math import *
```

A python dictionary stores a piece of data for each key. The keys in this case will be lists [x,y,z] of coordinates and the data will be the block types to draw. So we make a little utility function to draw the data in a dictionary:

```
def draw_data(x0,y0,z0,data):
 for key in data:      mc.setBlock(x0+key[0],y0+key[1],z0+key[2],data[x,y,z])
```

This loops through all the keys in the data, and draws the indicated block at the location, offsetting by (x0,y0,z0).

We now need to generate a knot. I used the cinquefoil formulas from here. This requires looping a parameter t from 0 to 2*pi, with small enough steps to ensure we don't have gaps. I used 10000 steps. Since this is done in-memory, and computers are fast, and overlapping blocks are only sent once to Minecraft, it's easier to easier to do more steps than to think how many is enough.

```
knot = {}
scale = 10 t = 0 while t < 2*pi:   x = int( scale * cos(2*t) * (3 + cos(5*t)) )   y = int( scale * sin(2*t) * (3 + cos(5*t)) )   z = int( scale * sin(5*t) )   knot[x,y,z] =
block.GOLD_BLOCK   t += 2*pi / 10000
```

This puts the data in the knot dictionary. Note that we need to round off the x, y and z coordinates with the int() function. That's the magic behind the overlap removal: when the rounded coordinates are the same, only one block is stored in the dictionary (dictionaries store only one value per key). You can resize the knot by changing the scale, though you may need to change the step size if you make it too big.

Finally we need to draw the knot:

```
mc = minecraft.Minecraft.create(server.address)
playerPos = mc.player.getPos() draw_data(playerPos.x,playerPos.y + 5 * scale,playerPos.z,knot)
```

Here we offset the knot upward by 5 * scale.

The above code is in **knot.py**.

The knot would look better if the rope were thicker. There are many ways one can do that. An inefficient way, but easy since computers are fast these days, is just to draw a little ball instead of a point at each pixel. To do that, first I make a little utility function to add a ball to a python dictionary:

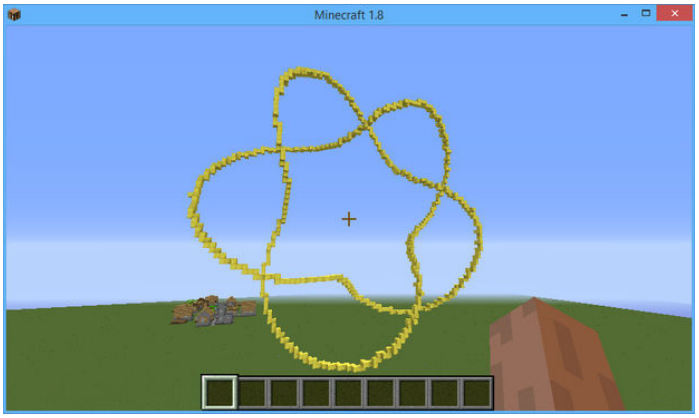
```
def ball(x0,y0,z0,r,block,data):
 for x in range(-r,r):     for y in range(-r,r):       for z in range(-r,r):         if (x**2 + y**2 + z**2 <= r**2):             data[x0+x,y0+y,z0+z] = block
```

This uses the inequality method to fill in a ball at (x0,y0,z0), of radius r, in the dictionary dict with the specified block type.

Then just modify our knot-making while loop to make a ball instead of just adding an entry to the knot dictionary:

```
knot = {}
scale = 10 t = 0 while t < 2*pi:   x = int( scale * cos(2*t) * (3 + cos(5*t)) )   y = int( scale * sin(2*t) * (3 + cos(5*t)) )   z = int( scale * sin(5*t) )
ball(x,y,z,4,block.GOLD_BLOCK,knot)   t += 2*pi / 10000
```

The result is in **knot2.py** in the sample scripts.



Step 9: Additional resources

The best resource for programming Minecraft in python is the Adventures in Minecraft book.

The author of the book also has a lot of good information on the stuffaboutcode.com site. In particular, he has an API reference here. The Raspberry Jam Mod (and mcpiapi mod) supports just about everything that the Raspberry Juice server plugin does.

The real-time working analog clock in the picture above is from one of the stuffaboutcode scripts (sitting above one of my donuts): **/python stuffaboutcode_clock**

I am planning to teach a summer middle/high school gifted class using python and Minecraft. Suggestions for neat projects for kids ar

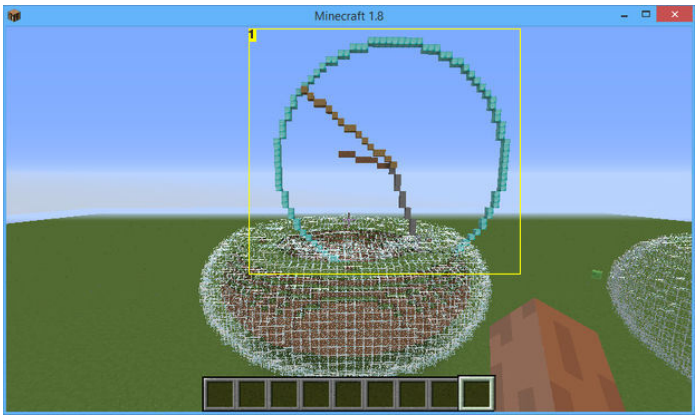


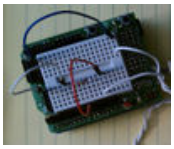
Image Notes
1. real-time working analog clock from stuffaboutcode.com, in stuffaboutcode_clock.py

Related Instructables

Synthesizer Awesome by JoeyPython

Real World Minecraft by sajingeo

Fully Automatic "Unlimited" Snow Farm by Arthak



simpleTweet_01 python by pdxnat

Arduino Minecraft Interface by GeckoScraps

How to install Python packages on Windows 7 by pdxnat

Comments

1 comments **Add Comment**

tomatoskins says:
I never knew that you could write scripts for minecraft! So cool!

Mar 29, 2015. 10:26 AM **REPLY**