

Exporting 3D objects from Processing

Zachary Steinberg

18 Oct 2021

(Warning: If you're reading this now, this is going to be relevant only in the last few weeks of the semester!)

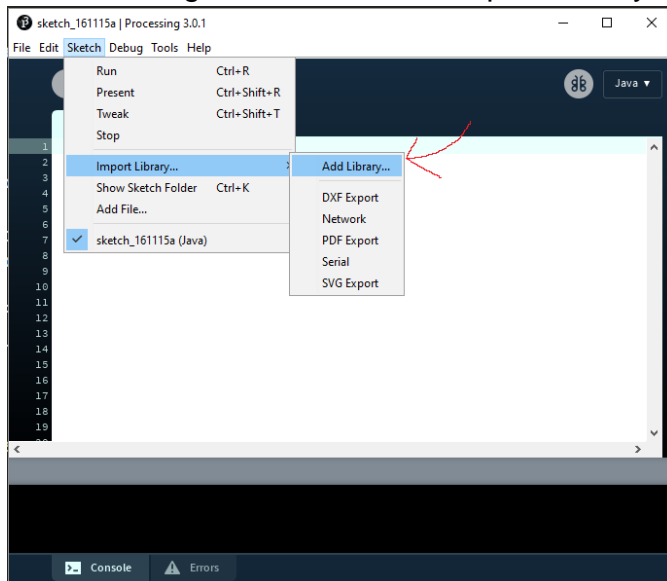
Towards the end of the semester, we'll learn Processing can draw 3D things. We also have a 3D printer, which can make those 3D things in real life! But to do that, we need to tell Processing to turn your 3D drawing into a .obj file that the 3D printing software can understand.

Step 0: Open processing

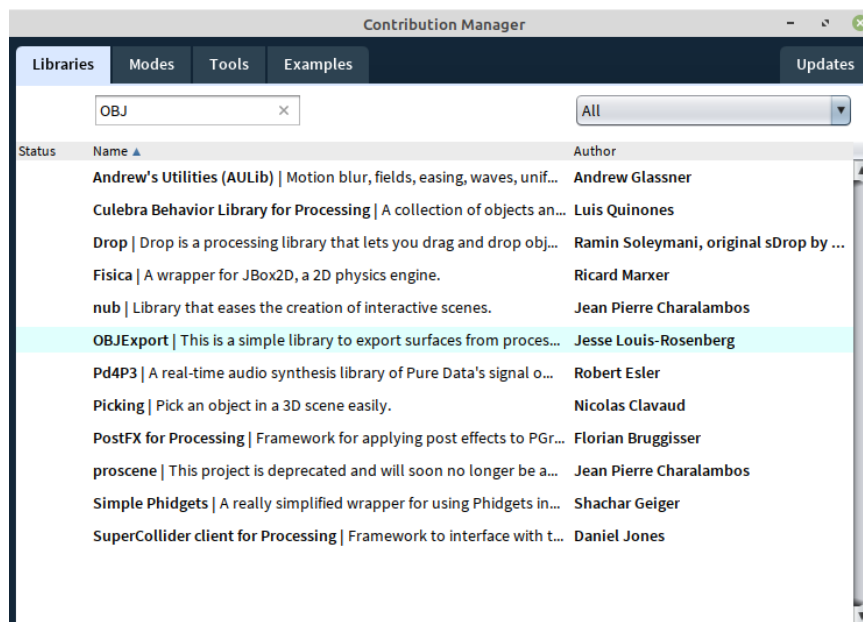
Hopefully you know how to do this by now.

Step 1: Install the OBJExport Library

In Processing, Go to Sketch -> Import Library -> Add Library



This will open up a window which shows you lots of libraries. We want to add OBJExport. Click on the “Libraries” tab in the top left, and search for “OBJExport” or “OBJ”.



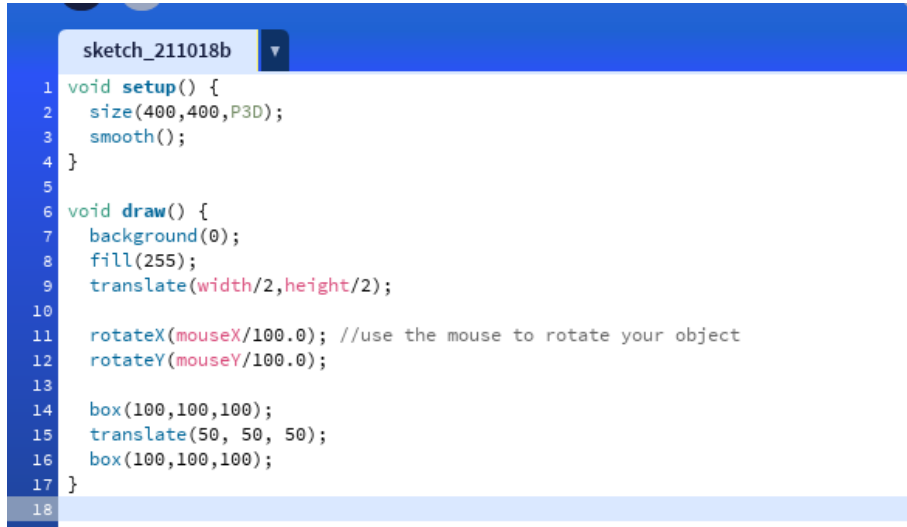
Click on OBJExport in the list, by Jesse Louis-Rosenberg, then click “Install” in the bottom right corner.

Once you’ve done this, close the Contribution Manager window.

From the main Processing window, go to Sketch -> Import Library again, but don’t click anything. If you see “OBJExport” on the bottom of the menu, it worked.

Step 2: How to use the OBJExport library

Let's say we have a program which makes a 3D object we want to export. Here's one, for example.



```
sketch_211018b ▼
1 void setup() {
2   size(400,400,P3D);
3   smooth();
4 }
5
6 void draw() {
7   background(0);
8   fill(255);
9   translate(width/2,height/2);
10
11   rotateX(mouseX/100.0); //use the mouse to rotate your object
12   rotateY(mouseY/100.0);
13
14   box(100,100,100);
15   translate(50, 50, 50);
16   box(100,100,100);
17 }
18
```

Step 2.1: From Processing, click Sketch -> Import Library. Since you completed Step 1, you should see an “OBJExport” entry in the menu. (If OBJExport isn’t there, see Step 1.) Click on OBJExport.

- This will add the line “import nervoussystem.obj.*;” to the top of your code. This line is what tells processing to know about the OBJ exporting code.

Step 2.2: Add this line to your code, BEFORE you start calling any drawing commands:

```
beginRecord("nervoussystem.obj.OBJExport", "filename.obj");
```

- This line of code will create a file named filename.obj. This function works like saveFrame in that you can replace “filename.obj” with “myobject.obj” or “someobject.obj” or whatever other filename you want, and it will change the name of the file that’s saved.

Step 2.3: Change “filename.obj” to something which better describes your object and add your initials on the front, like “MMShovel.obj”.

Step 2.3: Add this line to your code AFTER your drawing commands:

```
endRecord();
```

- My example now looks like this:

```
exportedbox
1 import nervoussystem.obj.*;
2
3 void setup() {
4   size(400,400,P3D);
5   smooth();
6 }
7
8 void draw() {
9   background(0);
10  fill(255); //note: 3D printers will ignore fill().
11  translate(width/2,height/2);
12
13  beginRecord("nervoussystem.obj.OBJExport", "MMShovel.obj"); // added
14
15  //rotateX(mouseX/100.0);
16  //rotateY(mouseY/100.0);
17
18  box(100,100,100);
19  translate(50, 50, 50);
20  box(100,100,100);
21
22  endRecord(); //added!
23 }
24
```

Step 2.4: Save your code somewhere, then run your code with `beginRecord()` and `endRecord()` in place.

Step 2.5: Just like `saveFrame`, open the folder where your code is saved, and you should see a new `.obj` file there!



Step 3 (Optional): If you want to make your code more organized, I suggest putting the code to draw your 3D object in a new function, like this:

```
void draw(){
  background(0);
  translate(width/2,height/2);
  //rotateX(mouseX/100.0); //commented out for recording
  //rotateY(mouseY/100.0);

  beginRecord("nervoussystem.obj.OBJExport", "MMShovel.obj");
  drawMy3DThing(); //you program this function
  endRecord();
}

void drawMy3DThing(){
  //3D drawing code goes here...
}
```

This way it's easier to see where to put beginRecord() and endRecord().