

# OpenScad examples

# Table of Contents

1. Document information .....	1
2. Objects .....	1
2.1. Object - Knurl .....	1
2.2. Object - Library-container .....	3
2.3. Object - RPi_zero_mount .....	8
2.4. Object - Test .....	11
2.5. Object - VacuumPreAmplifierBase .....	12
2.6. Object - buttonBack .....	15
2.7. Object - case .....	16
2.8. Object - fastener .....	19
2.9. Object - fridgeDoorInterimHandle .....	22
2.10. Object - geoTest .....	23
2.11. Object - ikeabung .....	24
2.12. Object - internal-volume .....	25
2.13. Object - lampRing .....	26
2.14. Object - midletoninset .....	28
2.15. Object - spool .....	31
2.16. Object - tesa .....	33
3. To do .....	35

# 1. Document information

## *Links to Document*



Document online



Document Source



Document PDF

test repo for building openscad files into different outputs

This is still work in progress but can already build a png and stl of each scad file in the opescad directory.

See the online or pdf versions for the images as the readme is really only the source and right now is not WYSIWYG!

## 2. Objects

### 2.1. Object - Knurl

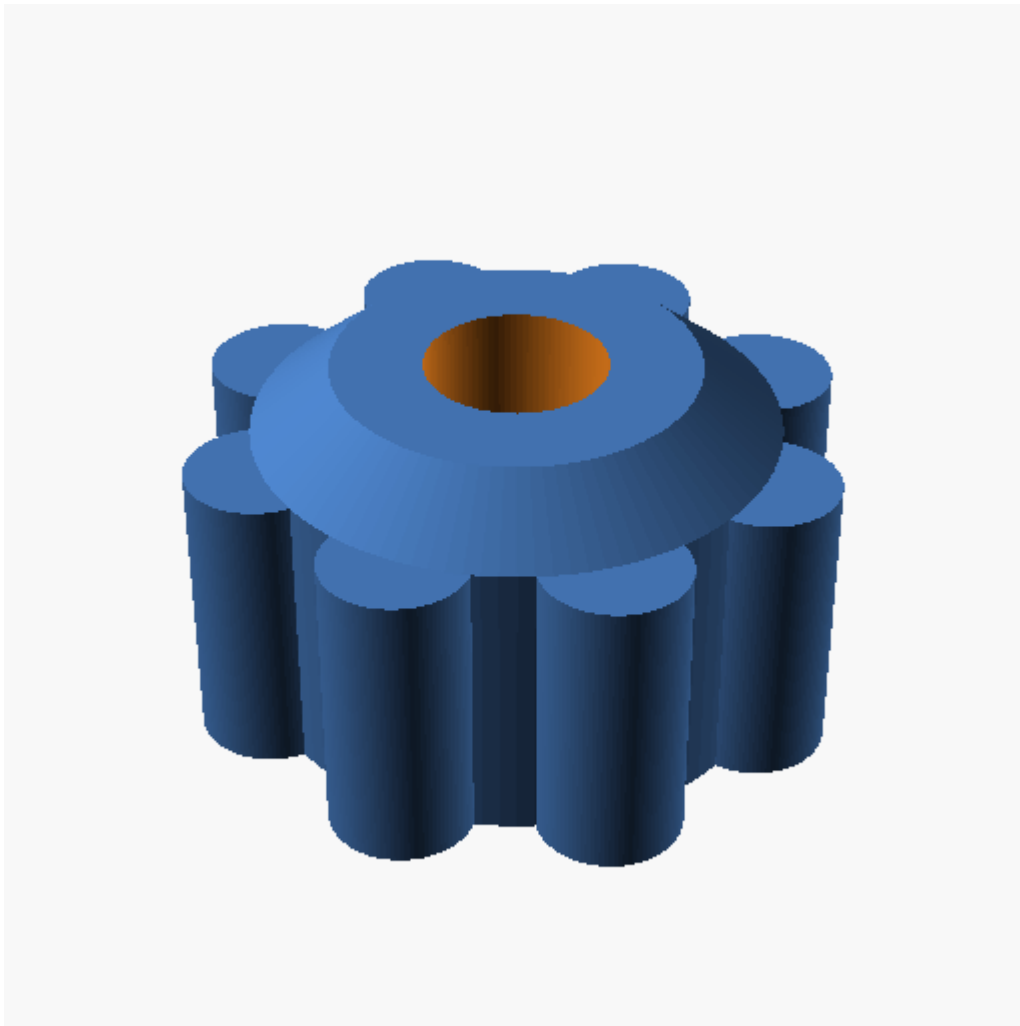


Figure 1. image

Listing 1. Openscad source

```
diam=9;
diamOut=14.4;
holeDiam=5;
height=8.2;
$fn=100;
knurlNum=8;
knurlInc=360/knurlNum;
knurlDiam=4;
insetHeight=5;
insetDiam=5;
totalHeight=10;

module knob() {
difference(){
union(){
cylinder(h=height, d=diamOut);
translate([0,0,height]){
```

```
        cylinder(h=totalHeight-height,r1=7.2,r2=5);
    }
}
translate([0,0,-.5])
    cylinder(h=totalHeight+1, d=holeDiam);
translate([0,0,-.5]){
    cylinder(h=totalHeight-insetHeight+.5,d1=holeDiam+5,d2=holeDiam);
}
}

for(i=[0: knurlInc: 360]){
    rotate([0,0,i])
        translate([diamOut/2,0,0])
            cylinder(h=height, d=knurlDiam);
}
}

knob();
```

## 2.2. Object - Library-container



Figure 2. image

Listing 2. Openscad source

```

module nibLeft(x,y,z,nibR) {
    translate([x,y/10,0]) cylinder(h=z,r=nibR);
    translate([x,9*(y/10),0]) cylinder(h=z,r=nibR);
}
module nibRight(x,y,z,nibR) {
    translate([0,y/10,0]) cylinder(h=z,r=nibR);
    translate([0,9*(y/10),0]) cylinder(h=z,r=nibR);
}
module nibBottom(x,y,z,nibR) {
    translate([x/10,0,0]) cylinder(h=z,r=nibR);
    translate([9*(x/10),0,0]) cylinder(h=z,r=nibR);
}
module nibTop(x,y,z,nibR) {
    translate([x/10,y,0]) cylinder(h=z,r=nibR);
    translate([9*(x/10),y,0]) cylinder(h=z,r=nibR);
}
module containerOpenLid(x,y,z,rimThick,bottomThick,nibYN,nibR) {
    rimR=rimThick/2;

```

```
//all 8 corners defined first
//corners should be AROUND the contained cube defined by x y z
corner000=[0,0,0];
corner0=[-rimR,-rimR,-(rimR+bottomThick)];
corner0x=[x+rimR,-rimR,-(rimR+bottomThick)];
corner0y=[-rimR,y+rimR,-(rimR+bottomThick)];
corner0xy=[x+rimR,y+rimR,-(rimR+bottomThick)];
corner0z=[-rimR,-rimR,z];
corner0xz=[x+rimR,-rimR,z];
corner0yz=[-rimR,y+rimR,z];
corner0xyz=[x+rimR,y+rimR,z]; //draw the debug contents
translate([0,0,-bottomThick]) cube([x,y,bottomThick]);
module corner0() {
    translate(corner0) sphere(r=rimR);
}
module corner0x() {
    translate(corner0x) sphere(r=rimR);
}
module corner0y() {
    translate(corner0y) sphere(r=rimR);
}
module corner0xy() {
    translate(corner0xy) sphere(r=rimR);
}
module corner0z() {
    translate(corner0z) sphere(r=rimR);
}
module corner0xz() {
    translate(corner0xz) sphere(r=rimR);
}
module corner0yz() {
    translate(corner0yz) sphere(r=rimR);
}
module corner0xyz() {
    translate(corner0xyz) sphere(r=rimR);
}
if (nibYN=="nibY") {
    nibBottom(x,y,z,nibR);
    nibLeft(x,y,z,nibR);
    nibRight(x,y,z,nibR);
    nibTop(x,y,z,nibR);
}
union(){
    //floor
    hull(){
        corner0();
        corner0x();
        corner0y();
        corner0xy();
    }
}
```

```

    //left
    hull(){
        corner0();
        corner0z();
        corner0y();
        corner0yz();
    }
    //right
    hull(){
        corner0x();
        corner0xy();
        corner0xyz();
        corner0xz();
    }
    //top
    hull(){
        corner0y();
        corner0yz();
        corner0xyz();
        corner0xy();
    }
    //bottom
    hull(){
        corner0();
        corner0z();
        corner0x();
        corner0xz();
    }
}

// example
//caseRim=1.5;
//$fn=100;
//odH=10;
//odW=156;
//odD=73;
//odJSH=6;
//#containerOpenLid(odW,odD,odH,caseRim,odJSH-caseRim,"nibY",.6);
//cube([odW,odD,odH]);

module containerVertSlot(x,y,z,rimThick,bottomThick,nibYN,nibR) {
    rimR=rimThick/2;
    //all 8 corners defined first
    //corners should be AROUND the contained cube defined by x y z
    corner000=[0,0,0];
    corner0=[-rimR,-rimR,-(rimR+bottomThick)];
    corner0x=[x+rimR,-rimR,-(rimR+bottomThick)];
    corner0y=[-rimR,y+rimR,-(rimR+bottomThick)];
    corner0xy=[x+rimR,y+rimR,-(rimR+bottomThick)];

```



```

corner0z=[-rimR,-rimR,z];
corner0xz=[x+rimR,-rimR,z];
corner0yz=[-rimR,y+rimR,z];
corner0xyz=[x+rimR,y+rimR,z]; //draw the debug contents
translate([0,0,-bottomThick]) cube([x,y,bottomThick]);
module corner0() {
    translate(corner0) sphere(r=rimR);
}
module corner0x() {
    translate(corner0x) sphere(r=rimR);
}
module corner0y() {
    translate(corner0y) sphere(r=rimR);
}
module corner0xy() {
    translate(corner0xy) sphere(r=rimR);
}
module corner0z() {
    translate(corner0z) sphere(r=rimR);
}
module corner0xz() {
    translate(corner0xz) sphere(r=rimR);
}
module corner0yz() {
    translate(corner0yz) sphere(r=rimR);
}
module corner0xyz() {
    translate(corner0xyz) sphere(r=rimR);
}
if (nibYN=="nibY") {
    nibLeft(x,y,z,nibR);
    nibRight(x,y,z,nibR);
}
union(){
    //floor
    hull(){
        corner0();
        corner0x();
        corner0y();
        corner0xy();
    }
    //left
    hull(){
        corner0();
        corner0z();
        corner0y();
        corner0yz();
    }
    //right
    hull(){

```

```
        corner0x();
        corner0xy();
        corner0xyz();
        corner0xz();
    }
}
}
// example
//caseRim=1.5;
//$fn=100;
//odH=10;
//odW=15;
//odD=73;
//odJSH=6;
//containerVertSlot(odW,odD,odH,caseRim,odJSH-caseRim,"nibY",.6);
//cube([odW,odD,odH]);

if (library) {} else {
    echo("trying to compile a library!");
    linear_extrude(height = 4) {
        text("trying to compile a library!");
    }
}
```

## 2.3. Object - RPi\_zero\_mount



Figure 3. image

Listing 3. Openscad source

```
/* [Base] */
//type = 1; //[1:"Hexagon Grid",2:"Skeleton"]

/* [Hidden] */
$fn = 32;
zero_x = 64;
zero_y = 29;
zero_z = 1.5;

mounts_z = 8.5;
mounts_radius = 2.1;
screw_holes = 2.6;
screw_holes_radius = 1.5;
screw_holes_depth = 10.7;

base_x = zero_x - 2*3.0;
base_y = zero_y - 2*3.0;
base_z = zero_z;
```

```

mount_x = zero_x/2 - screwholes;
mount_y = zero_y/2 - screwholes;
mount_z = zero_z + mounts_z;
screwhole_base_z = mount_z - screwholes_depth;

module baseplate(){
    translate([-zero_x/2+3,-zero_y/2+3,0])
        minkowski(){
            cube([base_x,base_y,base_z/2]);
            cylinder(r=3.0,h=base_z/2);
        }
}

module mounts(){
    translate([0,0,0]) cylinder(r=3.0,h=mount_z);
    translate([-mount_x, -mount_y, 0]) cylinder(r=mounts_radius,h=mount_z);
    translate([-mount_x, +mount_y, 0]) cylinder(r=mounts_radius,h=mount_z);
    translate([+mount_x, -mount_y, 0]) cylinder(r=mounts_radius,h=mount_z);
    translate([+mount_x, +mount_y, 0]) cylinder(r=mounts_radius,h=mount_z);
}

module hexagon (radius=8,latticeWidth=8,latticeLength=16,spacing=1,height=2){
    linear_extrude(height) {
        for(j = [0:latticeWidth-1]) {

translate([(sqrt(3)*radius)+spacing)/2*(j%2),sqrt((pow(((sqrt(3)*radius)+spacing),2))-
(pow((((sqrt(3)*radius)+spacing)/2,2))*j,0)) {
            for(i = [0:latticeLength-1]) {
                translate([(sqrt(3)*radius*i)+spacing*i,0,0]) {
                    rotate([0,0,30]) {
                        circle(radius, $fn = 6);
                    }
                }
            }
        }
    }
}

module hex_border(){
    difference(){
        baseplate();
        holes();
        translate([0,0,-.01]) scale([0.9,0.8,1.02]) baseplate();
    }
}

module holes(){
    translate([0,0,screwhole_base_z+0.4]) {
        translate([0,0,0]) cylinder(r=screwholes_radius*1.5,h=screwholes_depth);
    }
}

```

```

        translate([-mount_x, -mount_y, 0])
cylinder(r=screwholes_radius, h=screwholes_depth);
        translate([-mount_x, +mount_y, 0])
cylinder(r=screwholes_radius, h=screwholes_depth);
        translate([+mount_x, -mount_y, 0])
cylinder(r=screwholes_radius, h=screwholes_depth);
        translate([+mount_x, +mount_y, 0])
cylinder(r=screwholes_radius, h=screwholes_depth);
    };
}

module result(){
    difference(){
        translate([-2.5, -base_y/2, 0]) cube([5, base_y, base_z]);
        translate([0, 10, -3]) cylinder(d=1.5, h=10);
        translate([0, -10, -3]) cylinder(d=1.5, h=10);
        holes();
    }
    translate([0, 0, 0])
    hex_border();
    difference(){
        translate([0, 0, 0]) cylinder(r=3.0, h=mount_z);
        holes();
    }
    difference(){
        mounts();
        holes();
    }
    difference(){
        baseplate();
        holes();
        translate([-zero_x/2-5, -zero_y/2+1.5, -0.1]) hexagon();
    }
}

difference(){
    result();
    translate([0, 10, -3]) cylinder(d=1.5, h=10);
    translate([0, -10, -3]) cylinder(d=1.5, h=10);
}

```

## 2.4. Object - Test

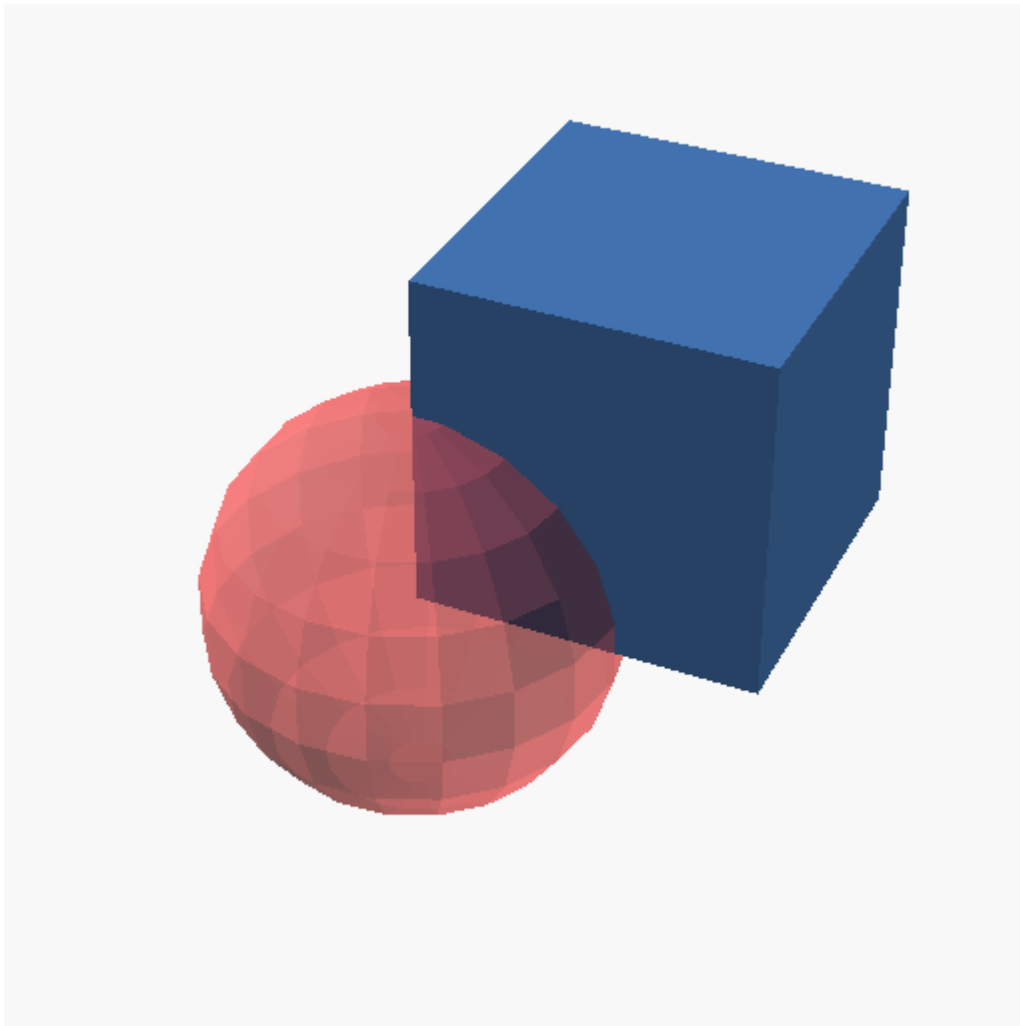


Figure 4. image

Listing 4. Openscad source

```
cube([10,10,10]);  
#sphere(d=12);
```

## 2.5. Object - VacuumPreAmplifierBase

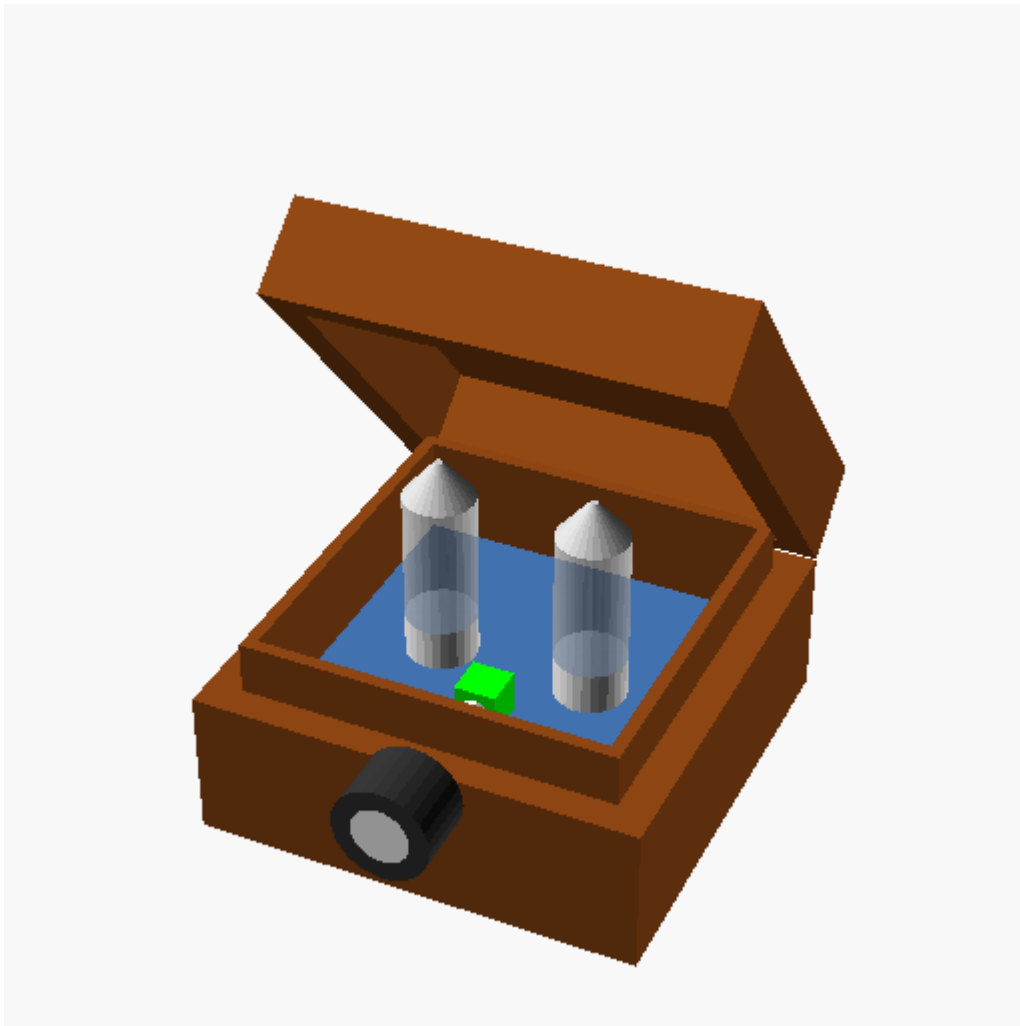


Figure 5. image

Listing 5. Openscad source

```
box1X=105.5;  
box1Y=106;  
box1Z=36;  
box1BaseH=3;  
  
box2X=89;  
box2Y=91;  
box2Z=45.5;  
  
box3X=83;  
box3Y=85;  
box3Z=50;  
  
lidX=box1X;  
lidY=box1Y;  
lidZ=23;  
lidDepth=20.3;  
lidStampR=20;
```

```

lidHingeAngle=50;
lidAnimZ=0;

preampBoardX=77;
preampBoardY=66;
preampBoardZ=1.5;
preampTubeR=17/2;
preampTubeH=42;
preampTubeBaseH=10;
preampTubeTipH=51;
preampTubeC=[200/255,200/255,200/255];
preampKnobR=11.5;
preampKnobH=16;
preampAxleH=29;

brown=[139/255,69/255,19/255];
gold=[255/255,215/255,0/255];
Blue=[0/255,0/255,200/255];

module box(){
    color(brown)
    difference(){
        union(){
            cube([box1X,box1Y,box1Z]);
            translate([box1X/2-box2X/2,box1Y/2-box2Y/2,box1BaseH])
cube([box2X,box2Y,box2Z]);
        }
        translate([box1X/2-box3X/2,box1Y/2-box3Y/2,3])
cube([box3X,box3Y,box3Z]);
        // star the next line to see inside the box
        *translate([- .5, - .5, - .5]) cube([box1X+1,box1Y*.85+1,box1Z/2+1]);
    }
}

//lid
module lid(){
    color(gold) translate([(box1X/2),(box1Y/2),lidZ+.0001])
cylinder(h=1,r1=lidStampR,r2=lidStampR);
    color(brown) translate([0,0,0])
        difference(){
            translate([0,0,.001]) cube([lidX,lidY,lidZ]);
            translate([box1X/2-box2X/2,box1Y/2-box2Y/2,0])
cube([box2X,box2Y,lidDepth]);
        }
}

module tube () {
    union(){
        color(preampTubeC,.5) translate([0,0,preampTubeBaseH])cylinder(h=42-
preampTubeBaseH,r1=preampTubeR,r2=preampTubeR);
    }
}

```



```

        color([1,1,1])cylinder(h=preampTubeBaseH,r=preampTubeR);
        translate([0,0,preampTubeH]) color(preampTubeC)
cylinder(h=preampTubeTipH-preampTubeH,r1=preampTubeR,r2=1);
    }
}

translate([(box1X-box3X)/2,((box1Y-box3Y)/2)+(box3Y-preampBoardY)-
1,box1BaseH+21])
union() {
    //board
    cube([preampBoardX,preampBoardY,preampBoardZ]);
    //tubes
    translate([15+preampTubeR,15+preampTubeR,preampBoardZ]) tube();
    translate([52+preampTubeR,15+preampTubeR,preampBoardZ]) tube();
    //Volume Knob Base
    translate([38,0,preampBoardZ]) color([0,1,0]) cube([10,10,10]);
    //volume knob
    translate([43,-(preampKnobH+preampAxleH),preampBoardZ+5]) rotate([270,0,0])
    union(){
        difference() {
            color([50/255,50/255,50/255]) cylinder(h=preampKnobH,r=preampKnobR);
            translate([0,0,-.001]) cylinder(h=1,r=(preampKnobR/100)*60);
        }
        color([255/255,255/255,255/255])cylinder(h=1,r=(preampKnobR/100)*60);
        //knob axle
        translate([0,0,preampKnobH]) color([1,1,1])cylinder(h=preampAxleH,r=3);
    }
}
//draft base
translate([15,box1Y-((box1Y-box3Y)/2)-1,box1BaseH])
color([0,0,0])cube([8,1,21]);
//enclosure
box();
translate([box1X,box1Y,(box1Z)+lidAnimZ+.5]) rotate([lidHingeAngle,0,180])
lid();

```

## 2.6. Object - buttonBack

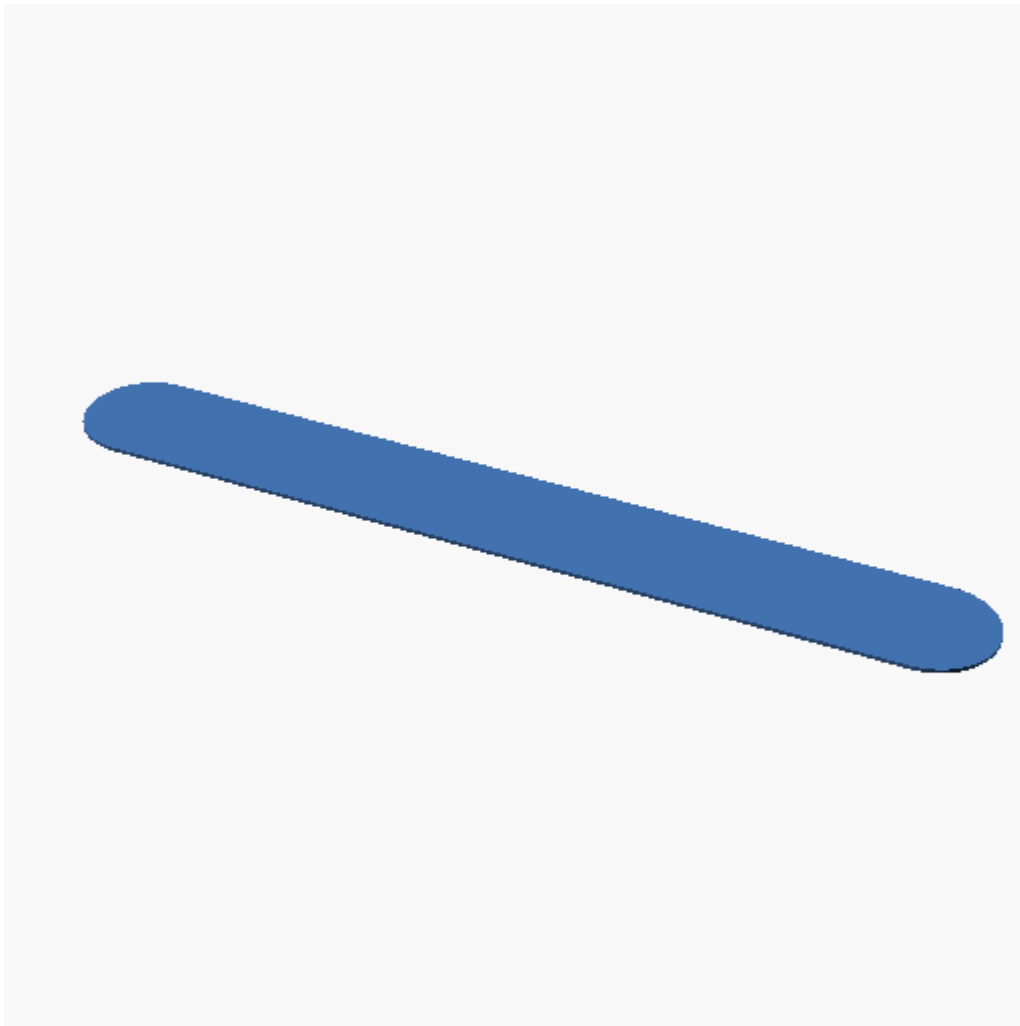


Figure 6. image

Listing 6. Openscad source

```
$fn=100;  
holeEndD=16.1;  
holeLength=120.1;  
buttonHolderHeight=.5;  
  
hull(){  
    cylinder(h=buttonHolderHeight,d=holeEndD);  
    translate([holeLength-holeEndD,0,0])  
    cylinder(h=buttonHolderHeight,d=holeEndD);  
}
```

## 2.7. Object - case

A case for an odroid handgeld console and accessories.

The edges are rounded and there are cutouts for the parts that protrude from the console.

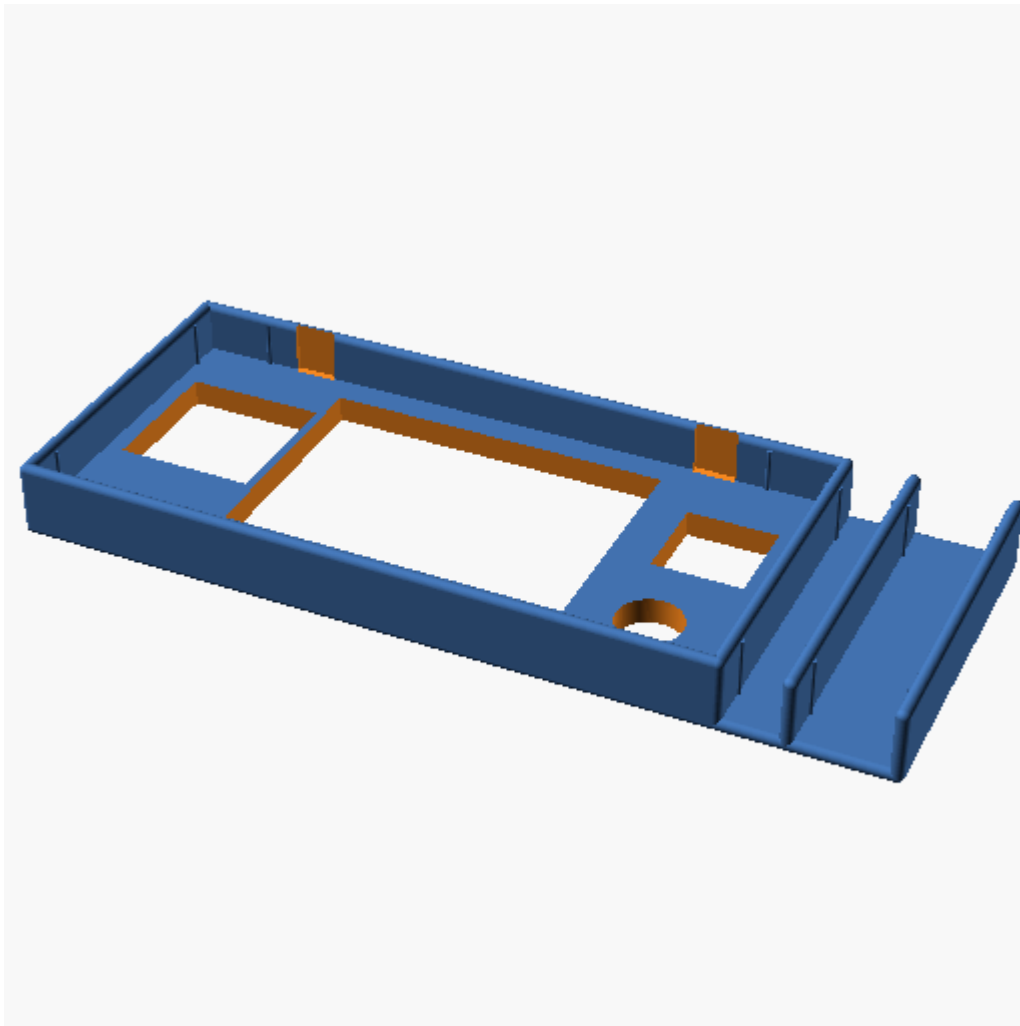


Figure 7. image

Listing 7. Openscad source

```
include <Library-container.scad>
caseRim=3;
holdersR=.7;
$fn=100;
kbD=82;
kbW=210;
kbH=7;
library=true;

odH=10;
odW=156;
odD=73;

odJSW=28-13;
odJSR=odJSW/2;
odJSoffX=13;
odJSoffY=11;
```

```

odJSH=6;
odBRW=118-38;
odBRD=12-5;
odBROffX=38;
odBROffY=5;
odCRW=28-7;
odCRD=58-37;
odCROffX=7;
odCROffY=37;
odTBW=152-120;
odTBD=61-27;
odTBOffX=120;
odTBOffY=27;
odTLOffX=23;
odTLOffY=odD;
odTLW=33-23;
odTLD=2;
odTH=20;
odTROffX=123;
odTROffY=odD;
odTRW=odTLW;
odTRD=odTLD;
odDPW=odBRW;
odDPD=67-14;
odDPoffY=14;
odDPoffX=odBROffX;

//the odroid travel case with cutouts for buttons etc
difference(){
    //the container itself
    translate([caseRim/2,caseRim/2,odJSH])
    containerOpenLid(odW,odD,odH,caseRim,odJSH-caseRim,"nibY",.6);
    offset=.01;
    //the cutouts
    translate([1.5,.75,-offset/2]) union() {
        translate([odW-odJSR*2-odJSoffX,odJSoffY,0]+[odJSR,odJSR,0])
        cylinder(h=odJSH+offset,r=odJSR);
        translate([odW-odBRW-odBROffX,odBROffY,0])
        cube([odBRW,odBRD,odJSH+offset]);
        translate([odW-odCRW-odCROffX,odCROffY,0])
        cube([odCRW,odCRW,odJSH+offset]);
        translate([odW-odTBW-odTBOffX,odTBOffY,0])
        cube([odTBW,odTBW,odJSH+offset]);
        translate([odW-odTLW-odTLOffX,odTLOffY,odJSH-.1])
        cube([odTLW,odTLD,odTH+offset]);
        translate([odW-odTRW-odTROffX,odTROffY,odJSH-.1])
        cube([odTRW,odTRD,odTH+offset]);
        translate([odW-odDPW-odDPoffX,odDPoffY,0])
        cube([odDPW,odDPD,odJSH+offset]);
    }
}

```

```
}  
//add on some slots for peripherals  
floorDepth=0;  
//microuter slot  
translate([caseRim/2+caseRim+odW,caseRim/2,floorDepth])  
    containerVertSlot(12,odD,odH+odJSH,caseRim,floorDepth-caseRim,"nibY",.6);  
//micro USB 3 Port Hub  
translate([caseRim/2+2*caseRim+odW+12,caseRim/2,floorDepth])  
    containerVertSlot(19.5,odD,odH+odJSH,caseRim,floorDepth-caseRim,"nibY",.6);
```

## 2.8. Object - fastener

This is a fastener for a writing Desk.

The idea is to add a magnet to hold it up and to print it so that it does not require a bearing.

- V1 is the first prototype for a first print test and fitting test
  - fits well and axle didn't print free so need update
- V2 added a better axle but didn't get printed
- V3 added a better cutout and is printed
  - The cutout is currently a dummy pending getting the axle to work to try it out with magnets taped into place
  - axle prints freely so moving on to screw holes, magnets, and covers
- V4 Added final OCD logo and screw caps etc.
  - Mounted and working.



Figure 8. image

Listing 8. Openscad source

```
$fn=100;
mainLength=50;
mainD=15;
mainH=10;
axleD=10;
axleDout=axleD+3;
ringH=2;

magnetX=17;
magnetY=5;
magnetZ=2;

module axle(xx1X,xx1Y) {
    translate([0,0,-xx1Y/2])cylinder(h=mainH+xx1Y,d=axleD+xx1X);
    translate([0,0,((mainH-ringH)/2)]) cylinder(h=ringH,d=axleDout+xx1X);
    translate([0,0,(mainH/2)-((axleDout-axleD)/2+ringH/2)])
cylinder(h=(axleDout-axleD)/2,d1=axleD+xx1X,d2=axleDout+xx1X);
    translate([0,0,(mainH/2)+(ringH/2)]) cylinder(h=((axleDout-
```

```

axleD)/2),d2=axleD+xxlX,d1=axleDout+xxlX);
}
module clip() {
    difference() {
        union(){
            hull(){
                cylinder(d=mainD,h=mainH);
                translate([mainLength,0,0]) cylinder(d=mainD,h=mainH);
            }
            translate([7,-3.5,mainH]) linear_extrude (height=1.5) {
                text("OCD",size=8);
            }
        }
    }
    //magnet
    translate([mainLength-magnetX,-magnetZ/2,(mainH-magnetY)/2+1])
cube([magnetX,magnetZ,magnetY+10]);
    //holder
    holderW=19;
    holderRin=33;
    holderRout=holderRin+holderW;
    difference(){
        translate([0,0,-.1]) cylinder(h=3+.1,r=holderRout);
        translate([0,0,-.11]) cylinder(h=3+.22,r=holderRin);
    }
}
}
module magnetCap(){
    //magnet cap
    difference(){
        cylinder(h=2.8,d=11);
        translate([0,0,-.1]) cylinder(h=2,d=10);
    }
}
module screwCap() {
    //screwcap axle
    cylinder(h=2,d=7.5);
    translate([0,0,2]) cylinder(h=1,d=axleD);
}

//add the clip
difference () {
    clip();
    axle(1,1);
}

//add the axle and drill a hole in it for a srew
difference(){
    axle(0,0);
    translate([0,0,-.05]) cylinder(h=mainH+.1,d=4);
    translate([0,0,mainH/2]) cylinder(h=(mainH/2)+.1,d=7.5);
}

```

```
//next two lines just a visual
//#translate([0,0,mainH+2]) screwCap();
//#translate([42,0,-.5]) magnetCap();
}
translate([0,-27,3]) rotate([0,180,0]) screwCap();
translate([0,-15,3]) rotate([0,180,0]) magnetCap();
```

## 2.9. Object - fridgeDoorInterimHandle



Figure 9. image

Listing 9. Openscad source

```
$fn=360;
Height=100;
Diameter=18;
HolePos=(Height/2);
HoleDiam=3;
HoleDepth=10;
difference () {
  hull() {
```



```

    translate([0,0,0])
        cylinder(h=1,d2=Diameter,d1=Diameter-2);
    translate([0,0,Height])
        cylinder(h=1,d1=Diameter,d2=Diameter-2);
}
translate([0,0,HolePos])
    rotate([90,0,0])
        cylinder(h=HoleDepth,d=HoleDiam);
}

```

## 2.10. Object - geoTest

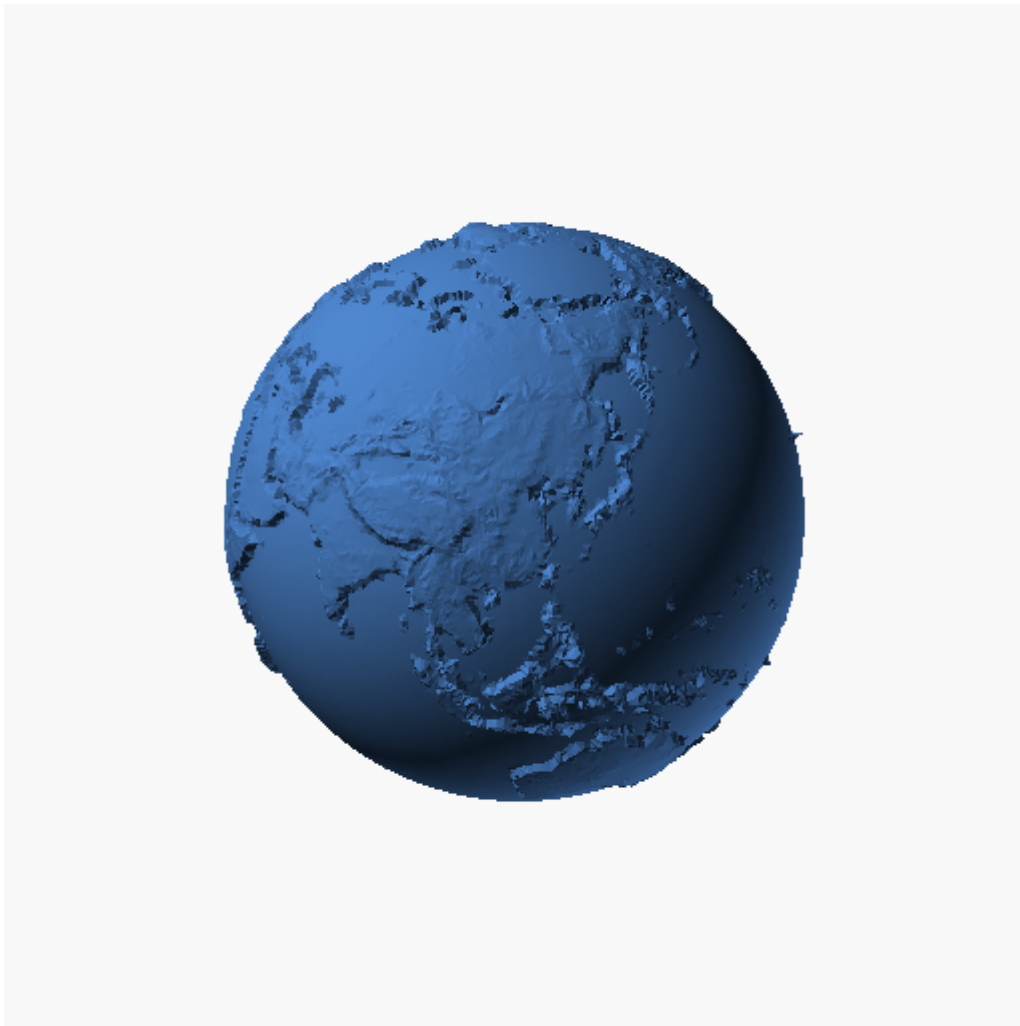


Figure 10. image

Listing 10. Openscad source

```

// Geody Planet 1 - SCAD
// Geody - https://www.geody.com/
// OpenSCAD - http://www.openscad.org/

wrrad=40; // Radius of the Planet

```

```
wrad=wwrad/20; // Radius of the Spot
wradp=wwrad-wrad/2; // Distance of the Spot from the center of the Planet
wres=50; // Resolution of the Spot

latx=48.782345; lonx=9.180819;

rotate(a=[0,0,270]) { import("geody_earthmap.stl", convexity=4); } // download
from https://www.geody.com/geody_earthmap.stl
// sphere(r=wwrad, $fn=wres); // Test Planet

translate([(-wradp)*cos(latx)*cos(lonx),(-
wradp)*cos(latx)*sin(lonx),wradp*sin(latx)]){sphere(r=wrad, $fn=wres,
center=true);}
```

## 2.11. Object - ikeabung

This was a replacement foot for an IKEA shelf.

The actual foot was screwed in with a bolt on the underside.

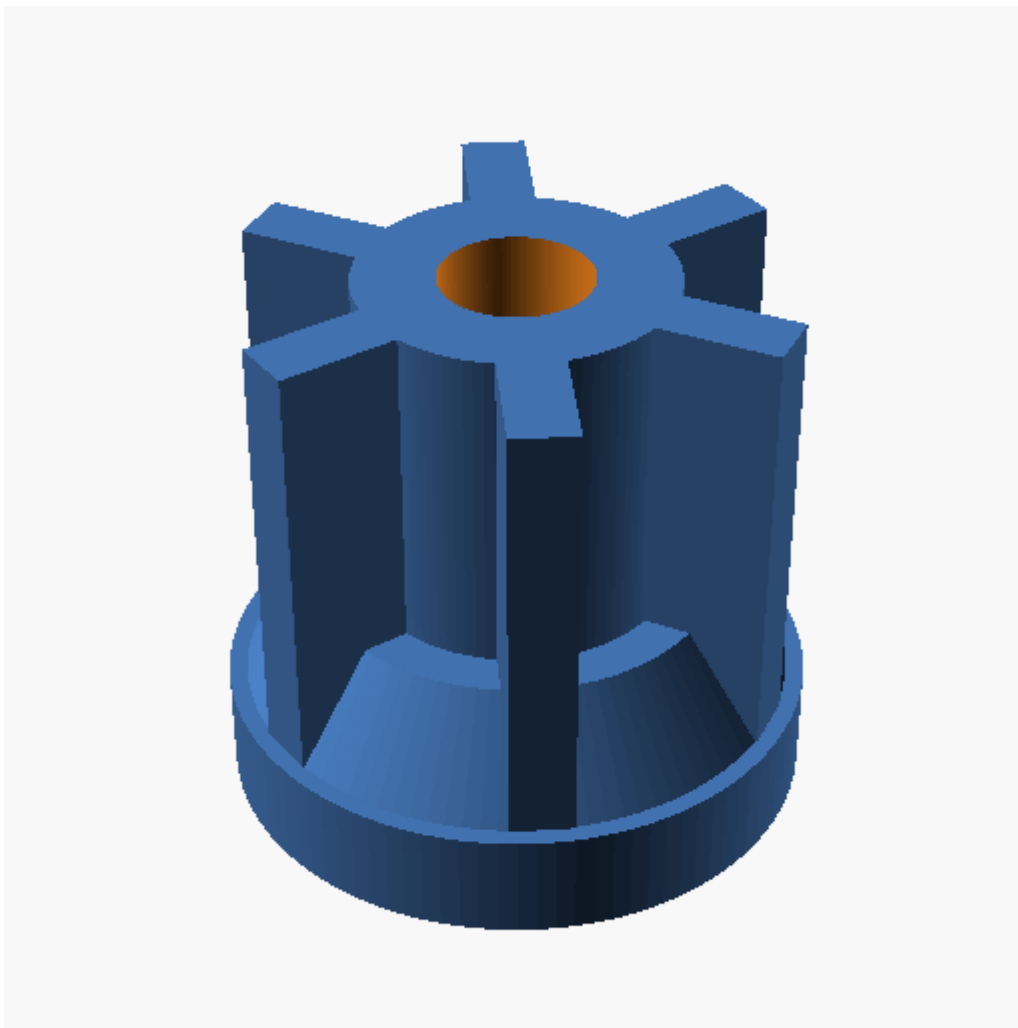


Figure 11. image

*Listing 11. Openscad source*

```
$fn=100;
totH=30;
baseH=6;
baseW=32;
wingW=3.5;
wingD=8;
centreD=17;

for (i = [0:360/6:360]) {
  rotate([0,0,i]) translate([((baseW-2)/2)-wingD,-wingW/2,baseH])
  cube([wingD,wingW,totH-baseH]);
}

difference(){
  union(){
    cylinder(h=totH,d=centreD);
    cylinder(h=6,d=32);
    translate([0,0,baseH]) cylinder(h=6,d1=baseW-2,d2=22);
  }
  translate([0,0,-.1]) cylinder(h=totH+.2,d=8.2);

  translate([0,0,-.1]) cylinder(h=8.1,d=15,$fn=6);
}
```

## 2.12. Object - internal-volume

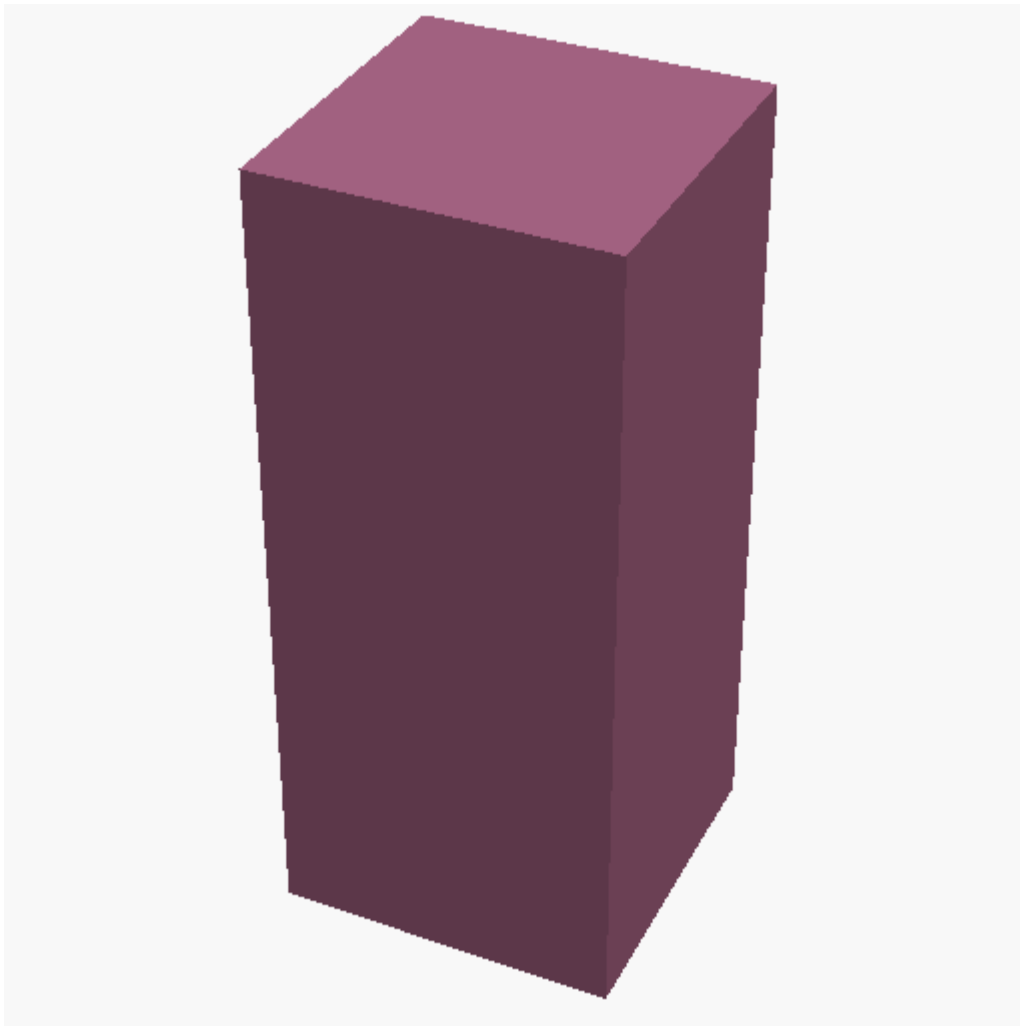


Figure 12. image

Listing 12. Openscad source

```
//inside of midleton wooden box with double doors  
height=260;  
width=111.1;  
depth=108.5;  
  
#cube([width,depth,height]);
```

## 2.13. Object - lampRing



*Figure 13. image*

*Listing 13. Openscad source*

```
//for LED lamps in ceiling in Howth
// the originals are wider and therefore the new ones need a spacer to cover the
hole
//colour is white
//led lamps are 105mm Diameter (4 lamps)

lampD=105;
lampH=2;
holeD=99;
coverD=125;
coverInD=99;
coverH=2;
coverRidgeW=5;
$fn=100;

//lamp
*color("white")
  translate([0,0,coverH])
```

```

        cylinder(h=2,d=lampD);

color("white") union(){
    difference(){
        cylinder(h=coverH,d=coverD);
        translate([0,0,-.1]) cylinder(h=coverH+.2,d=coverInD);
    }
    translate([0,0,coverH])
    difference(){
        cylinder(h=lampH,d=lampD+coverRidgeW);
        translate([0,0,-.1]) cylinder(h=lampH+.2,d=lampD+1);
    }
}
}

```

## 2.14. Object - midletoninset

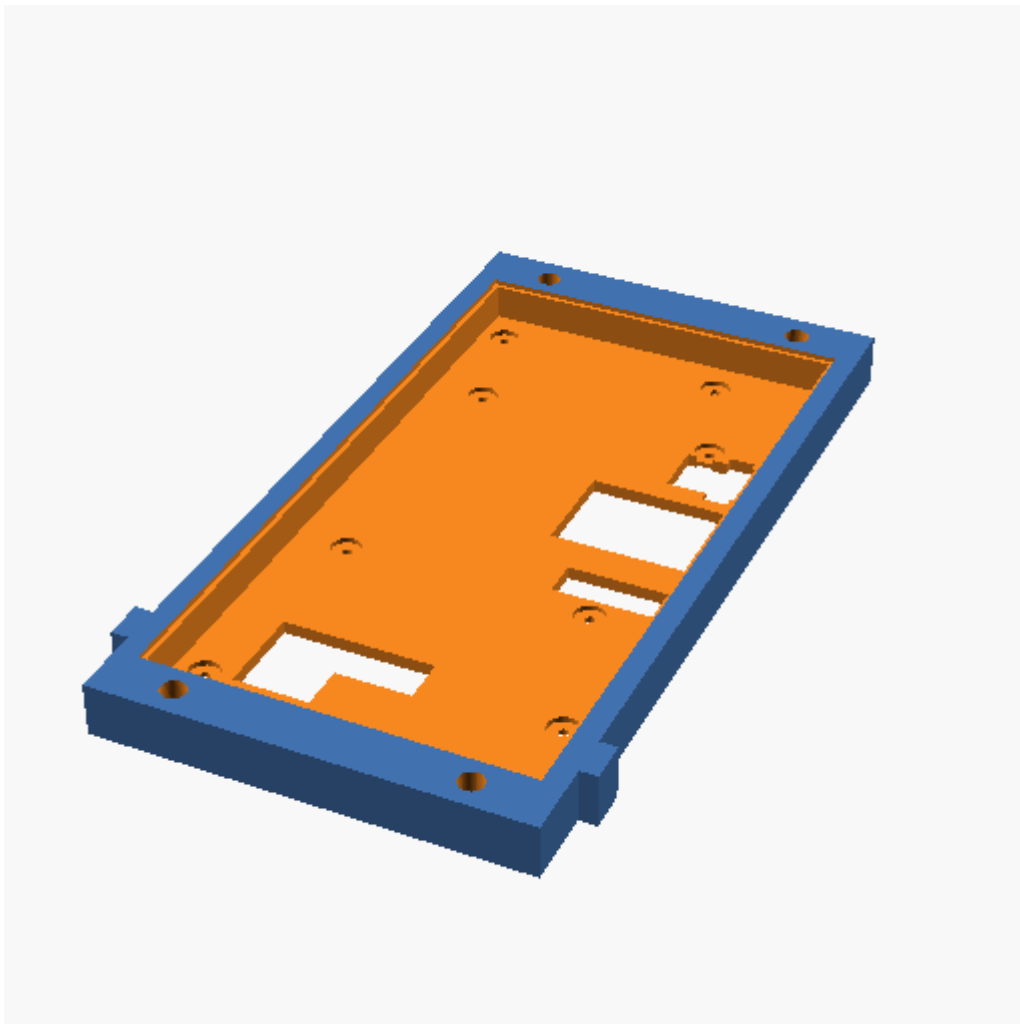


Figure 14. image

Listing 14. Openscad source

```
//This is an inlet for a whiskey presentation box from Midleton
```

```
$fn=50;
//Lower Notches
LowerNotchDepth=3.5;
LowerNotchLength=8;
LowerRNotchLengthOffset=15;
LowerLNotchLengthOffset=14.3;
module LLnotch(LowerLNotchLengthOffset){
    //Lower Left Notch
    translate([-LowerNotchDepth,LowerLNotchLengthOffset,0])
        cube([LowerNotchDepth,LowerNotchLength,BoxHeight]);
}
module LRnotch(LowerRNotchLengthOffset){
    //Lower Right Notch
    translate([BoxWidth,LowerRNotchLengthOffset,0])
        cube([LowerNotchDepth,LowerNotchLength,BoxHeight]);
}

//Variables for screen
ScreenTopY=75;
ScreenTopX=141;
ScreenTopZ=1;
ScreenEdge=1;
ScreenMaxDepth=7;
module waveshareHDMIscreen(wiggle){
    //full hd screen top face
    //and yes it has rounded corners but let's just start simple.
    union(){
        //Screen dimensions
        cube([ScreenTopY,ScreenTopX,ScreenTopZ+wiggle]);
        translate([ScreenEdge,ScreenEdge,-ScreenMaxDepth])
            cube([ScreenTopY-(2*ScreenEdge),ScreenTopX-
(2*ScreenEdge),ScreenMaxDepth+wiggle]);
        //connecting cable at the edge.
        translate([57,0,-ScreenMaxDepth]) cube([7,7,ScreenMaxDepth]);
        //USB for touch with offsetted connector - wiggle through
        translate([12,19,-ScreenMaxDepth-10])
            cube([30,9,ScreenMaxDepth+10]);
        translate([12,10,-ScreenMaxDepth-10])
            cube([15,12,ScreenMaxDepth+10]);
        //USB for power - wriggle through
        translate([65,95,-ScreenMaxDepth-10])
            cube([5,15,ScreenMaxDepth+10]);
        translate([57,97,-ScreenMaxDepth-10])
            cube([17,11,ScreenMaxDepth+10]);
        //HDMI connector - Wriggle through might not work... might have to make
hole larger
        translate([44,72,-ScreenMaxDepth-10])
            cube([30,20,ScreenMaxDepth+10]);
        //Audio?
        translate([51,56.75,-ScreenMaxDepth-4])
```

```

        cube([23,7.5,ScreenMaxDepth+4]);
//The screw holes
Standoffs();
//The mounting holes for the displaycover
translate([0+10,0-5,-ScreenMaxDepth-6])
    cylinder(h=20,d=4.8);
translate([0+10,ScreenTopX+5,-ScreenMaxDepth-6])
    cylinder(h=20,d=4.8);
translate([ScreenTopY-10,0-5,-ScreenMaxDepth-6])
    cylinder(h=20,d=4.8);
translate([ScreenTopY-10,ScreenTopX+5,-ScreenMaxDepth-6])
    cylinder(h=20,d=4.8);
    }
}
StandoffDepth=9;
StandoffSpace=1;
StandoffScrewHead=2;
module HolePeg(offset1){
    //standoff
    translate([0,0,-StandoffDepth+1]+offset1)
        cylinder(h=StandoffDepth-1,r=3.05);
    //screwshaft
    translate([0,0,-StandoffDepth-StandoffSpace+1]+offset1)
        cylinder(h=StandoffDepth+StandoffSpace-1,r=1);
    //Screw head
    translate([0,0,-StandoffDepth-StandoffSpace-StandoffScrewHead+1]+offset1)
        cylinder(h=StandoffScrewHead,r=3);
}
module Standoffs(){
    //Outside holes
    //one
    *HolePeg([6,9,0]);
    HolePeg([6.5,9.75,0]);
    //the rest
    HolePeg([69,22,0]);
    HolePeg([6,132.5,0]);
    HolePeg([53,132.5,0]);

    //inside holes
    HolePeg([11.5,52.5,0]);
    HolePeg([60.5,52.5,0]);
    HolePeg([60.5,110.5,0]);
    HolePeg([11.5,110.5,0]);
}

// Middleton box measurements
//Real total Height
//BoxHeight=61;
//Display inset Height
BoxHeight=10.5;

```



```
//testprint
//BoxHeight=8.5;
BoxWidth=83.8;
LowerPartLength=162.5;
//testing value
//LowerPartLength=50;
LowerPartWallThickness=1.5;
LowerPartFloorThickness=1.5;
module Displaymodule() {
    //Lower part of the box
    difference(){
        //Outercube
        cube([BoxWidth,LowerPartLength,BoxHeight]);
        //subtract for inner space

*translate([LowerPartWallThickness,LowerPartWallThickness,LowerPartFloorThicknes
s])
        cube([BoxWidth-2*LowerPartWallThickness,LowerPartLength,BoxHeight-
(2*LowerPartFloorThickness)]);
    }
    LLnotch(LowerLNotchLengthOffset);
    LRnotch(LowerRNotchLengthOffset);
}
//Displaymodule();
//Standoffs();
//wvshareHDMIscreen();

// put it all together
difference(){
    Displaymodule();
    //Screen
    translate([(BoxWidth-ScreenTopY)/2,(BoxWidth-ScreenTopY)/2+6,BoxHeight-
ScreenTopZ])
        wvshareHDMIscreen(.1);
    //for testprint only
    *translate([-10,10,2.5])cube([100,130,15]);
    *translate([10,-10,2.5])cube([65,160,15]);
}
//remove for print... only for animation
*translate([(BoxWidth-ScreenTopY)/2,(BoxWidth-ScreenTopY)/2+6,(BoxHeight-
ScreenTopZ)+30*(1-$t)]) wvshareHDMIscreen(0);
```

## 2.15. Object - spool

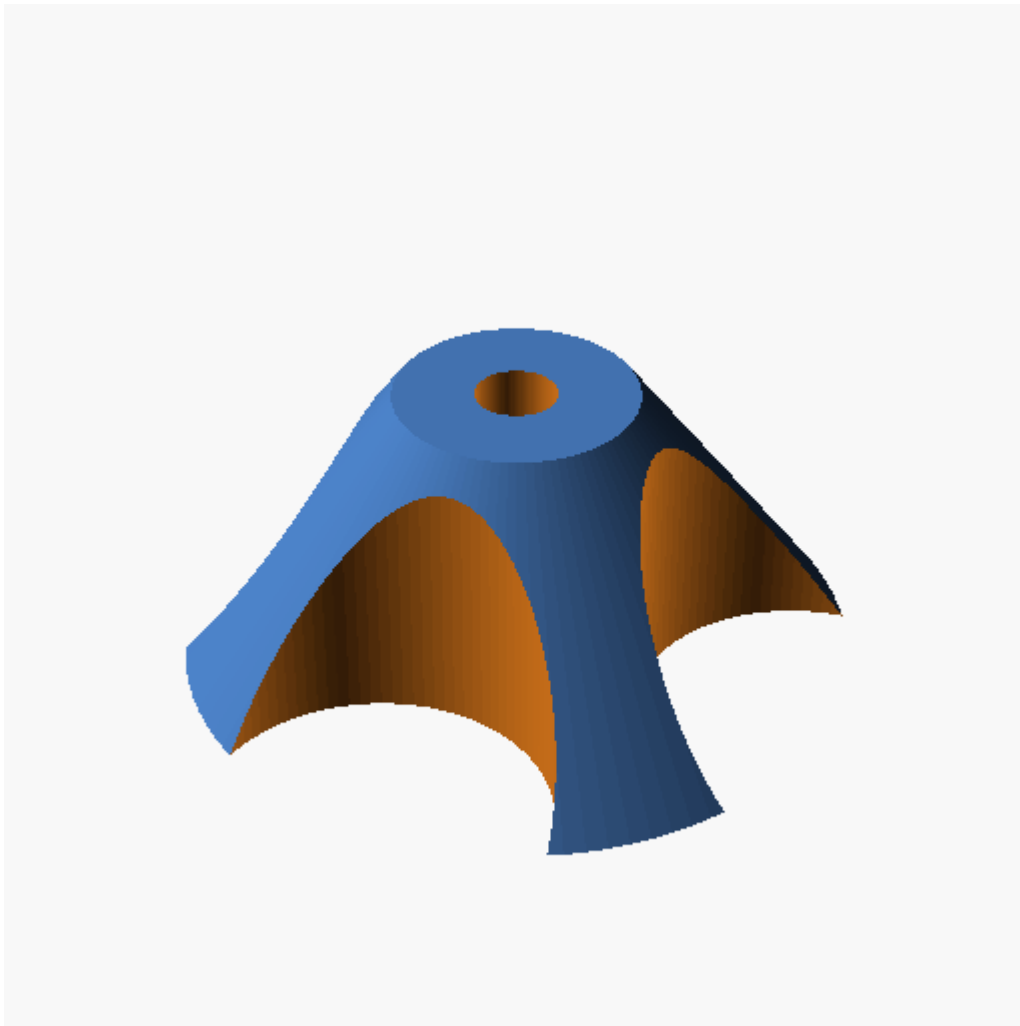


Figure 15. image

Listing 15. Openscad source

```

coneH=30;      //height of the cone
coneDin=25;    //smallest diameter of the cone
coneDout=70;   //widest diameter of cone
axleD=8;       //axle diameter of the axle for the 608 bearing - we'll add for
printer tolerance
$fn=100;       //make things round
bearingH=7;    //608 skateboard bearing height
bearingD=22;   //608 skateboard bearing diameter we'll add a millimeter or two
later to account for the fitting ring
fittingD=bearingD+7; //outer diameter of the fitting ring for the bearing
nubAngle=360/8; //the fitting nubs for the bearing at x degree rotation
printerRadTol=.2; //add this value to the radius
nubRad=.5;     //the nub radius for the bearing fitting ring

module cone(height,inD,outD) {
    cylinder(h=coneH , r2=(inD/2) , r1=(outD/2) );
}

```

```

module axle(height,diameter,tol) {
    translate([0,0,-.1]) cylinder(h=height,r=(diameter/2)+tol); //axle
}
module bearing(height,diameter,tol) {
    translate([0,0,-.1]) cylinder(h=height+.1,r=(diameter/2)+tol); //bearing
}

//subtract for quicker print
module removeCyls(bearingD,coneDout,coneH){
    translate([-((bearingD/2)+(coneDout/4)+4),0,-.1])
cylinder(h=coneH,r=coneDout/4);
    translate([(+(bearingD/2)+(coneDout/4)+4),0,-.1])
cylinder(h=coneH,r=coneDout/4);
    translate([0,+(bearingD/2)+(coneDout/4)+4),-.1])
cylinder(h=coneH,r=coneDout/4);
    translate([0,-((bearingD/2)+(coneDout/4)+4),-.1])
cylinder(h=coneH,r=coneDout/4);
}

module ring(inRad,outRad,height,tol) {
    difference(){
        cylinder(h=height,r=outRad+tol);
        translate([0,0,-.1]) cylinder(h=height+.2,r=inRad+tol);
    }
}

module fittingNubsCircle(nubRad,height,inRad,angle,tol) {
    rad=inRad+nubRad+tol;
    for (pos=[0:angle:360]) {
        *echo(pos);
        rotate ([0,0,pos]) translate([rad,0,0]) cylinder(h=height,r=nubRad);
    }
}

//
difference(){
    union(){
        difference(){
            cone(coneH,coneDin,coneDout);
            translate([0,0,-.1]) bearing(bearingH+.1,fittingD,printerRadTol);
            translate([0,0,-.1]) axle(coneH+.5,axleD,printerRadTol);
        }//
        ring( (bearingD/2)+nubRad, (fittingD/2) , bearingH , printerRadTol );
        fittingNubsCircle( nubRad , bearingH , bearingD/2 , nubAngle ,
printerRadTol );
    }//
    removeCyls(bearingD,coneDout,coneH);
}

```

## 2.16. Object - tesa

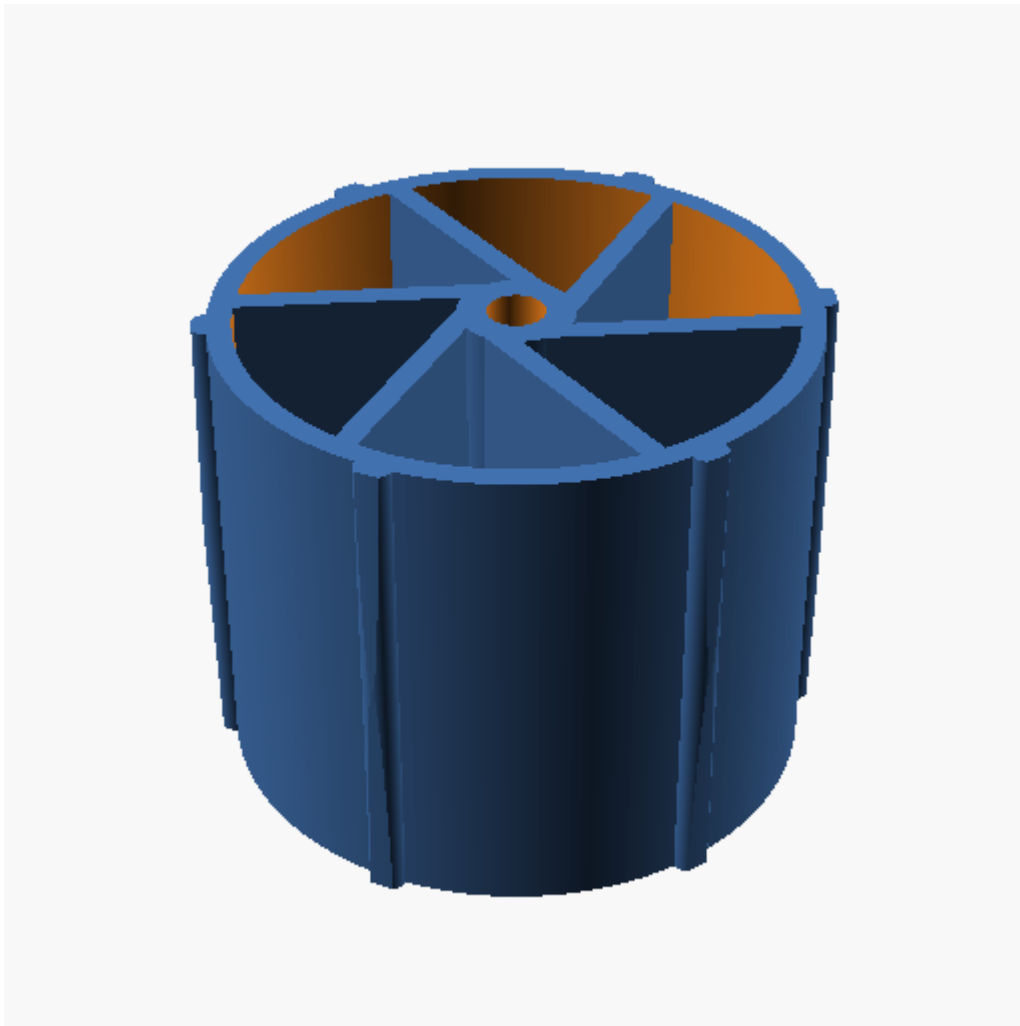


Figure 16. image

Listing 16. Openscad source

```
//Tesa roller ersatzroller
//celotape roller

$fn=360;
height=20;
outsideD=24.5;
outsideDepth=2;
axleD=2.4;
hubD=axleD*2;
nubR=.75;
module taper(){
difference(){
union(){
    translate([0,0,-.1]) cylinder(h=height+.2,d=outsideD+2+nubR);
}
union(){
    translate([0,0,-
.11])cylinder(h=height/2+.22,d1=outsideD+1.5*nubR,d2=outsideD+2*nubR);
```

```

    translate([0,0,height/2+.1])
cylinder(h=height/2+.1,d1=outsideD+2*nubR,d2=outsideD+1.5*nubR);
}
}
}
difference(){
union(){
    //outside
    difference(){
        cylinder(h=height,d=outsideD);
        translate([0,0,-.1])cylinder(h=height+.2,d=outsideD-outsideDepth);
    }
    //HUB
    difference(){
        cylinder(h=height,d=hubD);
        translate([0,0,-.1])cylinder(h=height+.2,d=axleD);
    }
    //nubs
    for (i = [0:5]) {
        translate([sin(360*i/6)*outsideD/2, cos(360*i/6)*outsideD/2, 0 ])
        rotate([0,0,0])cylinder(h = height/2, r=nubR);
    }
    for (i = [0:5]) {
        translate([sin(360*i/6)*outsideD/2, cos(360*i/6)*outsideD/2, height/2 ])
        cylinder(h = height/2, r=nubR);
    }
    //spokes
    for (i = [0:360/6:360]) {
        rotate([0,0,i])translate([1.2,0,0])cube([1,(outsideD/2)-(
axleD/2.4),height]);
    }
}
taper();
}

```

## 3. To do

Right now the github source is not perfect as the readme does not display the images when viewed in github.

Need to add further process steps for the images like meshlabserver to do further processing:

- glass rendering
- cleaning up the mesh
- Simplifying the mesh
- Stats

Need to add animation options.

- ☒ Need to add text display option for each item.

Need to add view parameters as options.