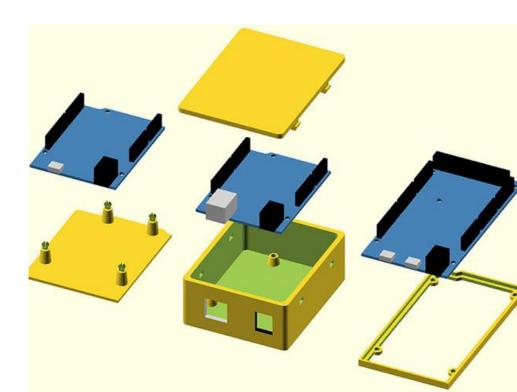
OpenScad

```
000
                                                                                          OpenSCAD - example009.scad
bodywidth = dxf_dim(file = "example009.dxf", name = "bodywidth");
fanwidth = dxf_dim(file = "example009.dxf", name = "fanwidth");
platewidth = dxf_dim(file = "example009.dxf", name = "platewidth");
fan_side_center = dxf_cross(file = "example009.dxf", layer = "fan_side_center");
fanrot = dxf_dim(file = "example009.dxf", name = "fanrot");
% linear_extrude(height = bodywidth, center = true, convexity = 10)
   import(file = "example009.dxf", layer = "body");
% for (z = [+(bodywidth/2 + platewidth/2),
        -(bodywidth/2 + platewidth/2)])
   translate([0, 0, z])
   linear_extrude(height = platewidth, center = true, convexity = 10)
       import(file = "example009.dxf", layer = "plate");
intersection()
   linear_extrude(height = fanwidth, center = true, convexity = 10, twist = -fanrot)
       import(file = "example009.dxf", layer = "fan_top");
   // NB! We have to use the deprecated module here since the "fan_side"
   // layer contains an open polyline, which is not yet supported
   // by the import() module.
   rotate_extrude(file = "example009.dxf", layer = "fan_side",
       origin = fan_side_center, convexity = 10);
```

- Why use it?
 - Free!
 - Large community: http://forum.openscad.org/
 - Easier to get started than other softwares;



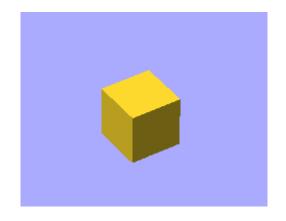
- Geometric Figures
 - Cube
 - Sphere
 - Cylinder
- Transformations
 - Translating
 - Scaling
 - Rotating
 - Mirroring

CSG

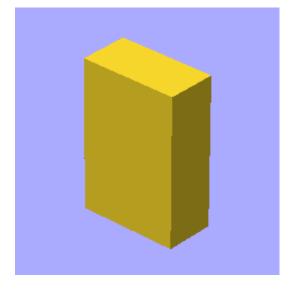
- Constructive Solid Geometry
 - Union
 - Difference
 - Intersection

```
Inton() {
  translate([0, 0, 120])
    difference() {
      union() {
        sphere(28);
        translate([0, 0, 20 * sin(30)])
          cylinder(30, 20 * cos(30), 0);
        translate([0, 0, 30 + 20 * sin(30)])
      //cut out slot
      rotate([45, 0, 0])
translate([-20, 0, 0])
cube([40, 5, 40]);
  // neck
  cylinder(120, 18, 12);
  cylinder(20, 35, 25);
  // collar
  translate([0, 0, 90])
    intersection() {
     cylinder(28, 20, 0);
     tronslate([8, 0, 7])
Viewport: translate = [ -12.30 3.25 29.79 ], rotate = [ 66.90 0.00 78.20 ], distance = 1590.52
```

- Cube
 - \$ cube([x,y,z]);
 - Cube([10,10,10]);

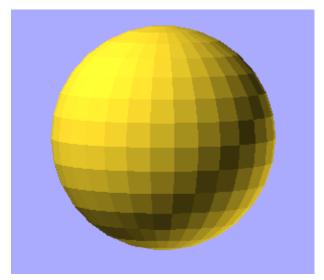


cube([10,20,30], center = true);



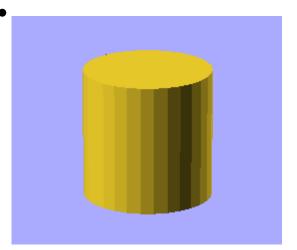
- Sphere
 - \$ sphere(r);
 - sphere(r=20); // already centered

•

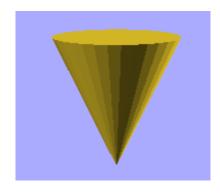


- Cylinder
 - \$ cylinder(h,r1,r2);
 - Cylinder(h=20,r=10);

Gy....a.c. (... 20). 20)



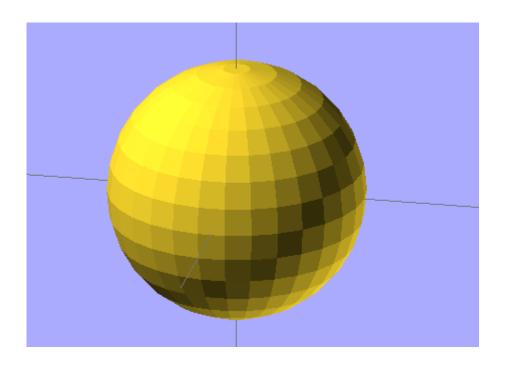
Cylinder(h=20,r=10, r2=0, center = true);



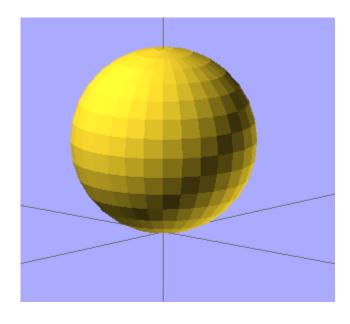
Transformations

Translation => Move to somewhere else;

• Sphere(r=20);



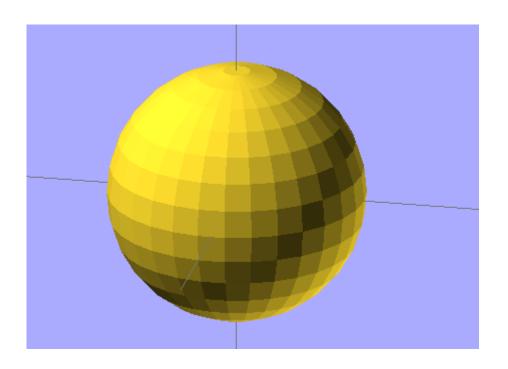
translate([0,0,20] Sphere(r=20);



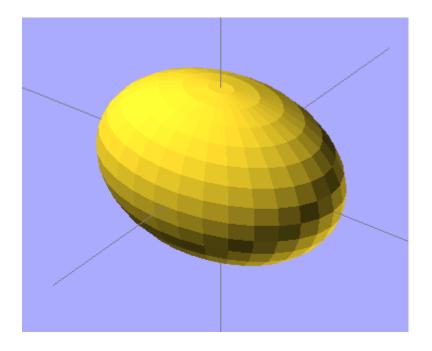
Transformation

Scaling=> multiply for some value;

• Sphere(r=20);



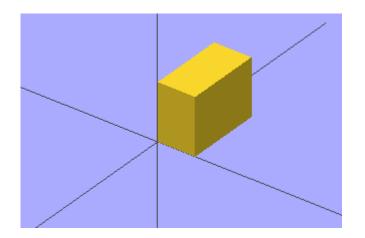
scale([1.5,1,1] Sphere(r=20);



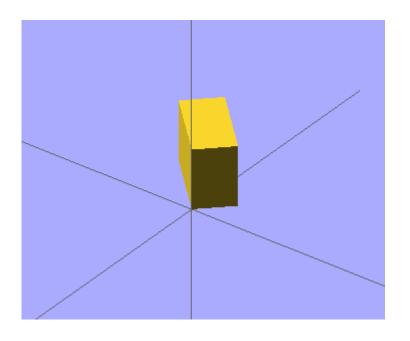
Transformation

Rotate=> change position in relation to an axis;

• cube([10, 20, 15]);

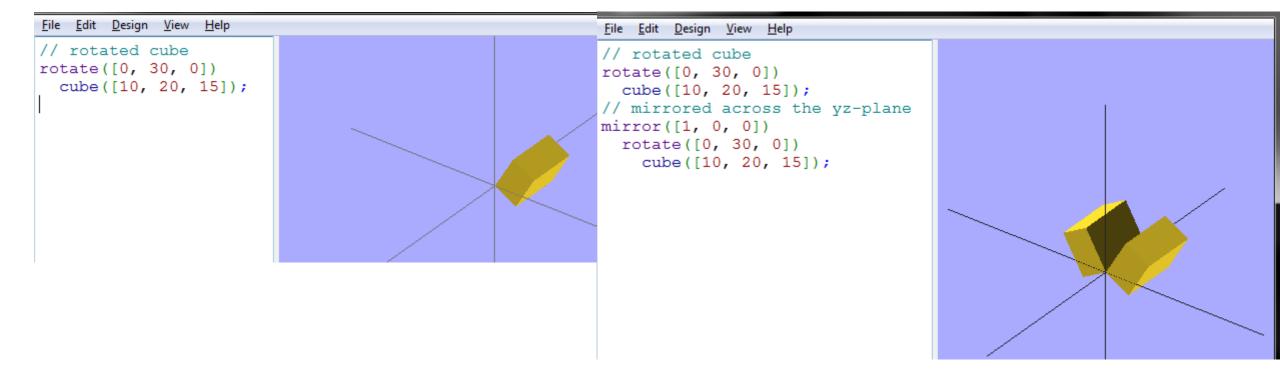


rotate([0,0,45] cube([10, 20, 15]);



Transformation

Mirror=> reflect in relation to an axis;



Constructive Solid Geometry Operations

• Union

```
union() {
  cylinder(h = 40, r = 10, center = true);
  rotate([90, 0, 0])
    cylinder (h = 40, r = 9, center = true);
}
```

Constructive Solid Geometry Operations

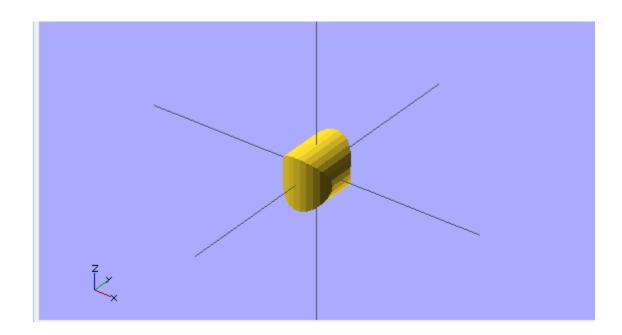
• Difference

```
difference() {
  cylinder(h = 40, r = 10, center = true);
  rotate([90, 0, 0])
   cylinder(40, r = 9, center = true);
}
```

Constructive Solid Geometry Operations

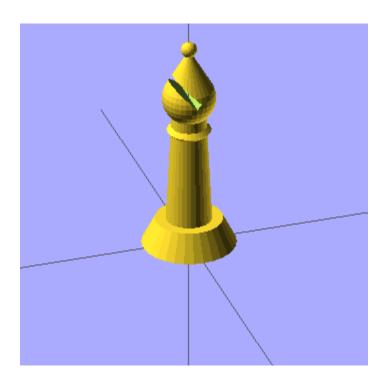
Intersection => What will overlap?

```
intersection() {
  cylinder(h = 40, r = 10, center = true);
  rotate([90, 0, 0])
    cylinder(40, r = 9, center = true);
}
```



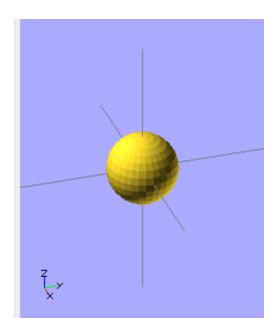
Got the basics?

- Practice, practice!
 - And one example



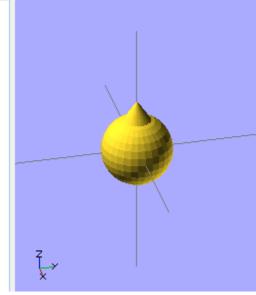
• Start with the "teardrop" shape;

sphere(r=20);



- Start with the "teardrop" shape;
- Add a cone to the sphere;

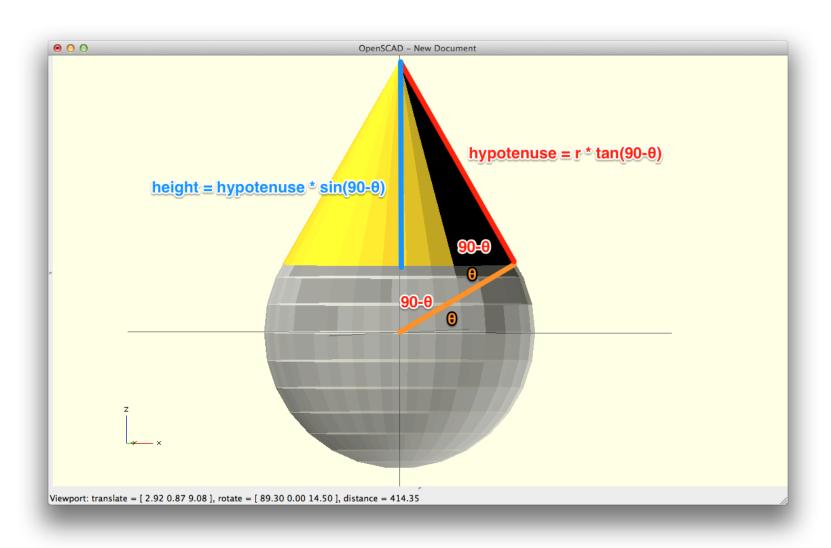
```
union() {
  | sphere(r = 20);
  cylinder(h = 30, r1 = 20, r2 = 0);
}
```



- Start with the "teardrop" shape;
- Add a cone to the sphere;
- Match the cone with the sphere

```
union() {
  sphere(r = 20);
  translate([0, 0, 20 * sin(30)])
   cylinder(h = 30, r1 = 20 * cos(30), r2 = 0);
}
```

Why sin(30)???



• Cut the piece (difference)

```
Elle Edit Design View Help

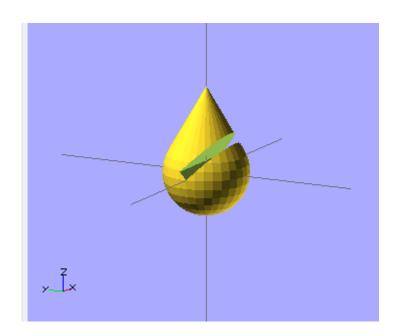
difference() {
    union() {
        sphere(r = 20);
        translate([0, 0, 20 * sin(30)])
        cylinder(h = 30, r1 = 20 * cos(30), r2 = 0);
    }
    cube([40, 5, 40]);
}
```

Correct the cut position

```
difference() {
  union() {
    sphere(r = 20);
    translate([0, 0, 20 * sin(30)])
      cylinder(h = 30, r1 = 20 * cos(30), r2 = 0);
  }
  translate([-20, 0, 0])
    cube([40, 5, 40]);
}
```

• Still missing the rotation!

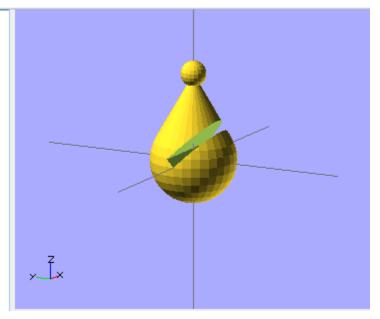
```
difference() {
  union() {
    sphere(r = 20);
    translate([0, 0, 20 * sin(30)])
        cylinder(h = 30, r1 = 20 * cos(30), r2 = 0);
  }
  rotate([45, 0, 0])
    translate([-20, 0, 0])
    cube([40, 5, 40]);
}
```



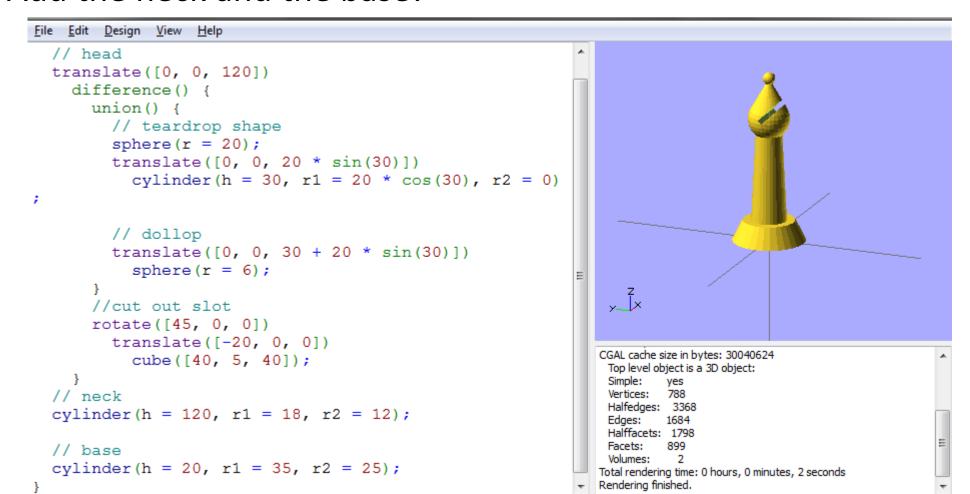
• Now, let's add a dollop on top. Since we know the height of the cone is 30, and we moved it up 20 * sin(30), we'll need to translate the dollop 30 + 20 * sin(30). We'll also comment the parts so we don't get confused.

```
difference() {
  union() {
    // teardrop shape
    sphere(r = 20);
    translate([0, 0, 20 * sin(30)])
      cylinder(h = 30, r1 = 20 * cos(30), r2 = 0);

    // dollop
    translate([0, 0, 30 + 20 * sin(30)])
      sphere(r = 6);
}
//cut out slot
rotate([45, 0, 0])
    translate([-20, 0, 0])
    cube([40, 5, 40]);
}
```



Add the neck and the base!

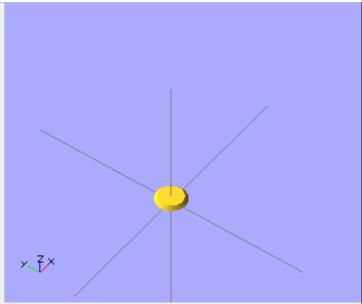


Making a fancy collar

```
cylinder(h = 20, r1 = 20, r2 = 0);
mirror([0, 0, 1])
  cylinder(h = 20, r1 = 20, r2 = 0);
```

Making a fancy collar

```
intersection() {
  cylinder(h = 20, r1 = 20, r2 = 0);
  translate([0, 0, 7])
    mirror([0, 0, 1])
    cylinder(h = 20, r1 = 20, r2 = 0);
}
```



Everything together!

```
// dollop
       translate([0, 0, 30 + 20 * sin(30)])
          sphere(r = 6);
     //cut out slot
     rotate([45, 0, 0])
       translate([-20, 0, 0])
          cube([40, 5, 40]);
// neck
cylinder (h = 120, r1 = 18, r2 = 12);
// base
cylinder (h = 20, r1 = 35, r2 = 25);
// collar
                                                                       Saved backup file: E:/Victor/My
translate([0, 0, 90])
                                                                       Documents/OpenSCAD/backups/unsaved-backup-hgqH1736.scad
  intersection() {
                                                                        Compiling design (CSG Tree generation)...
     cylinder(h = 20, r1 = 20, r2 = 0);
                                                                        Rendering Polygon Mesh using CGAL...
                                                                        PolySets in cache: 4
     translate([0, 0, 7])
                                                                        PolySet cache size in bytes: 87936
       mirror([0, 0, 1])
                                                                        CGAL Polyhedrons in cache: 72
                                                                       CGAL cache size in bytes: 31935080
          cylinder(h = 20, r1 = 20, r2 = 0);
                                                                         Top level object is a 3D object:
```