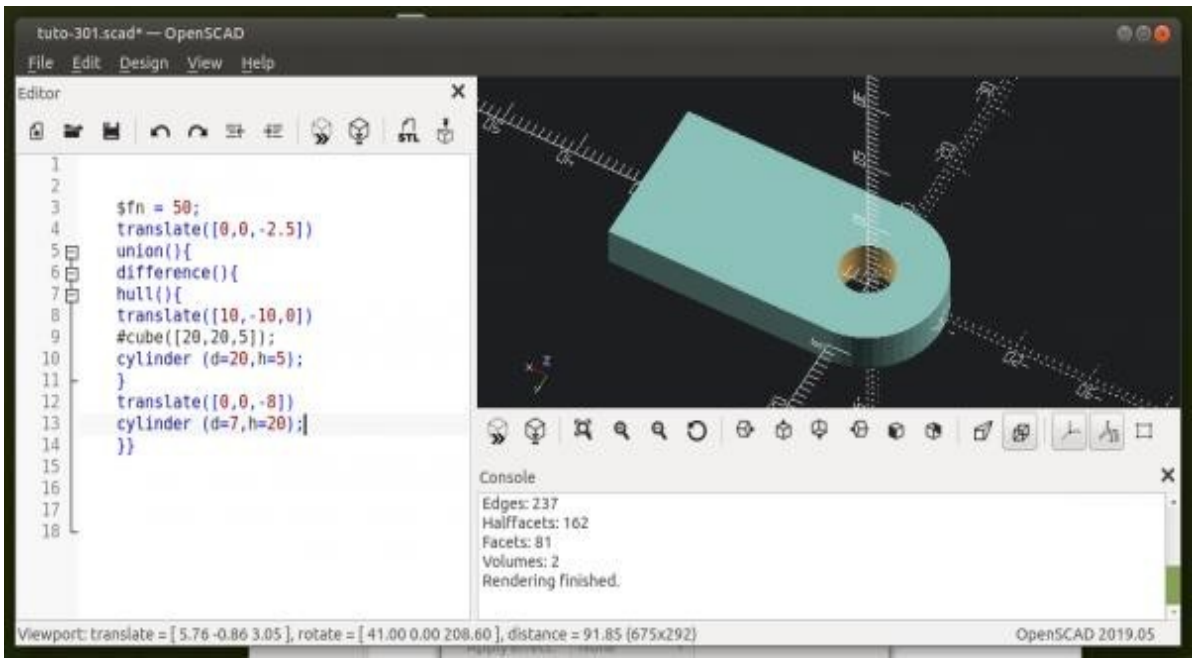
Brève introduction à la modélisation 3D avec OpenSCAD

Roberto Hamm – robotix.ah-oui.org

Mai 2021

L'interface de travail de OpensCAD

Les fenêtres : À gauche se trouve l'éditeur de code, à droite l'epace d'affichage tridimensionnel des objets et en dessous la console où s'inscrivent les actions réalisée par le programme. **Téléchargez OpenSCAD ici**



Les outils

1. Ouvrir un nouveau document OpenSCAD
2. Ouvrir un document existant
3. Sauvegarder le document en cours
4. Effacer le dernière action
5. Rétablir l'action effacée
6. Aligner tout à gauche
7. Créer de l'indentation
8. Prévisualiser
9. Créer le fichier STL
10. Sauvegarder le fichier STL
11. Lancer l'impression

A. Créer le fichier STL
B. Sauvegarder le fichier STL
C. Prévisualisation de l'ensemble du script
D. Zoomer
E. Dezomer
F. Vision globale 3D
G H I J K L. Différents angles de vue
M. Perspective
N. Vue orthogonale
O. Efface les coordonnées
P. Affiche le coordonnées
Q. Vue de axes du maillage

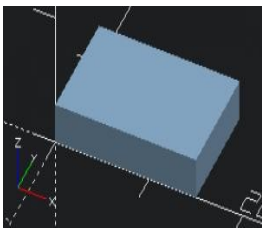
Fonctions essentielles

Les fonctions **cube**, **cylinder** et **sphere** permettent de créer les éléments de base de nos futures modélisations.

Toutes les valeurs correspondant aux axes X Y Z et pour le cylindre d=diamètre r=rayon et h=hauteur sont exprimés en millimètres.
Il est très important de veiller aux caractères suivants :
`( [ ] ) { } , ;`
un oubli ou erreur d'écriture bloquera le programme.

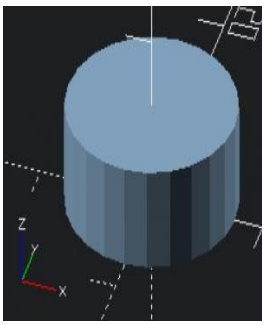
Les fonctions et commandes de openSCAD sont toujours écrites en minuscules et sont des termes en anglais (faciles à traduire en français).

Il est important de respecter les signe de ponctuation, parenthèses, crochets et accolades, présents dans les scripts. L'éditeur OpenSCAD souligne d'ailleurs les parenthèses, crochets et accolade d'ouverture ou fermeture, pour une meilleure visibilité. Le oublis ou erreurs sont marqués par un point rouge.



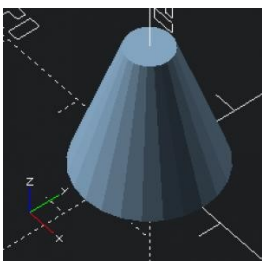
```
cube([10,20,3]);
```

La fonction **cube** sert à créer un parallélépipède dont les dimensions sont définies par des valeurs X, Y, et Z.



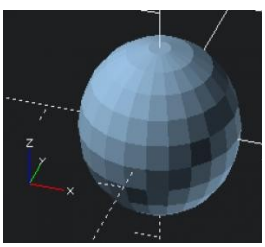
```
cylinder(d=15,h=5);
```

cylinder crée un cylindre de diamètre **"d"** et de hauteur **"h"**.
"d" peut re remplacé par **"r"**, dans ce cas il s'agit du rayon du cylindre.



```
cylinder(h=15,d1=15,d2=5);
```

En incluant **deux** valeurs de rayon ou de diamètre (**r1**, **r2** ou **d1** et **d2**) on peut créer de cônes.



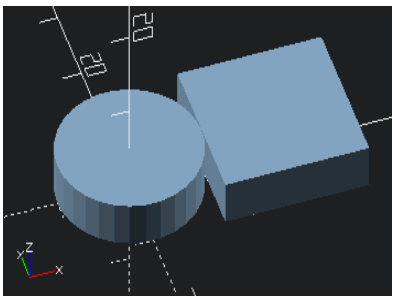
```
sphere(8);
```

a fonction sphere modélise un sphère deu diamètre indique entre les parenthèses.

Opérations de base : coordonnées X Y Z

```
cube([20,20,5]);
cylinder (d=20,h=5);
```

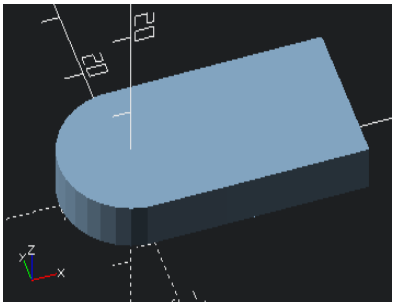
Pour commencer ce tutoriel nous allons créer 2 formes simples, un **cube** et un **cylindre**.



```
translate([10,-10,0])

cube([20,20,5]);
cylinder (d=20,h=5);
```

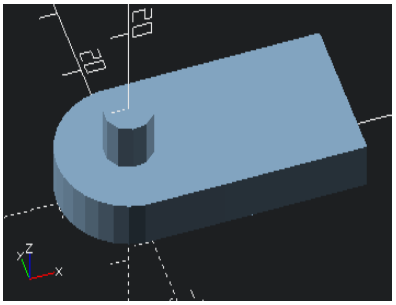
Ici nous déplaçons le cube avec l'instruction **translate ([x , y , z])** pour l'aligner avec le cylindre. Notez que l'instruction translate n'est pas terminée par un point-virgule car cette instruction s'adresse à la fonction cube qui se trouve directement après.



```
hull(){
translate([10,-10,0])

cube([20,20,5]);
cylinder (d=20,h=5);
}
```

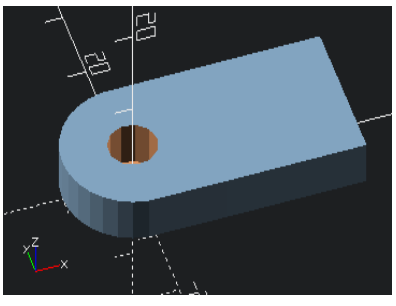
La fonction **hull()** englobe le cylindre et le le cube entre les accolades **{ }** et les fusionne. *Hull* en anglais signifie envelopper.



```
hull(){
translate([10,-10,0])

cube([20,20,5]);
cylinder (d=20,h=5);
}
cylinder (d=7,h=10);
```

Ici nous avons juste inclus un **cylindre** d'un diamètre de 7 mm et une hauteur de 10 mm, dans le centre de notre premier cylindre.



```
difference(){
hull(){
translate([10,-10,0])

cube([20,20,5]);
cylinder (d=20,h=5);
}
cylinder (d=7,h=10);
}
```

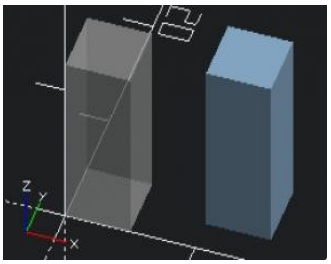
Ici nous incluons ge groupe hull() limité par ses accolades dans un nouveau groupe avec la commande **difference()** limité à son tour par ses acollades **{ }**. Cela à pour résultat de percer le grupe de **hull()** avec notre deuxième cylindre d'un diamètre 7 mm.

Modificateurs

Le modificateurs qui précèdent les fonctions, cube, cylindre, sphère ou celles que nous allons créer, servent à déplacer ou changer d'inclinaison les objets crées.

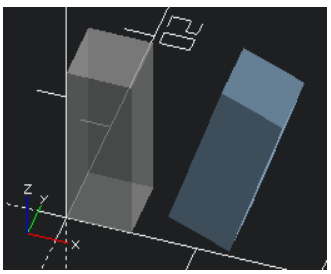
Les fonction précédées par **%** indiquent les positions dé départ des objets (le commandes de translation ne sont pas inscrites ici).

Important : Les modificateurs ne terminent pas par un "point virgule" car il se référent la fonction qui les suit (cube, cylinder, etc).



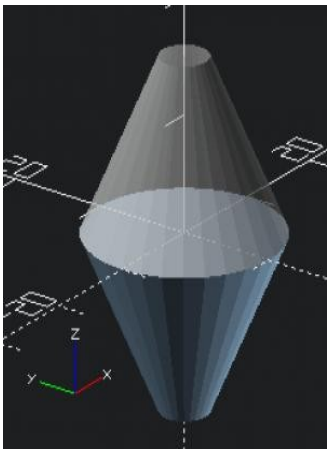
```
%cube([5,5,12]);
translate([10,2.5,0])
cube([5,5,12]);
```

translate déplace l'objet en fonction des coordonnées X, Y et Z. Ceci permet de positionner différents objets les uns par rapport aux autres.



```
%cube([5,5,12]);
rotate([0,20,0])
cube([5,5,12]);
```

rotate permet d'incliner l'objet selon une valeur indiquée en degrés. Dans notre exemple l'objet a été incliné à 20] sur son axe Y.

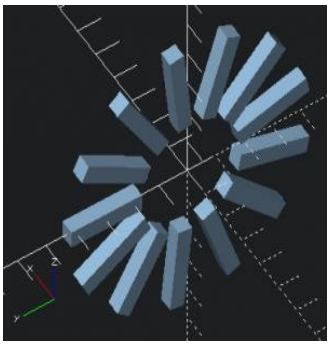


```
%cylinder(h=15,d1=20,d2=5)
;
mirror([0,0,1])
cylinder(h=15,d1=20,d2=5);
```

mirror permet de créer une réflexion (un effet miroir de l'objet) sur son axe Z.

Itérations

La "boucle" for permet de réaliser des itérations, c'est dire de répéter l'inclusion 'un me élément plusieurs fois et au besoin dans des positions différentes.



```
nb = 13;

for (i = [1:nb])
{
rotate([360/nb*i, 0, 0])
translate([0, 30, 0])
cube([5,25,5],center=true);

}
```

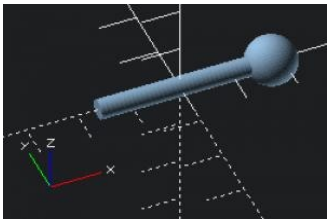
La boucle **for(i = [1:nb]){ ..... }** répétera un élément **nb** fois.
Dans cet exemple il s'agit de dessiner 13 fois le **cube([5,25,5]);** autour d'un point central imaginaire. La commande **rotate** déplace le cube sur la distance de 60°/13 et la commande **translate** déplace le cube de 30 mm vers l'extérieur.

La boucle "**for**" offre de nombreuses applications et peut être utilisée de diverses manières. Pour approfondir nous vous conseillons de consulter le **"CheatSheet"** de OpenSCAD et plus précisément la description des utilisations de la boucle **"for"** sur **ce lien**

Modules

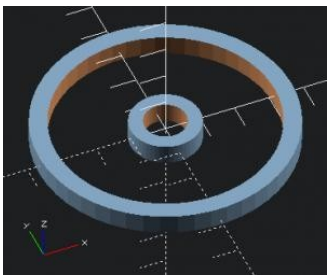
Les modules sont de courts scripts qui modélisent différentes parties pour construire un modèle 3D plus complexe. Comme une fonction ils peuvent être appelé plusieurs fois.

Dans cet exemple mous allons modéliser une barre à roue utilisant 3 modules.



```
module rayons(){
rotate([0,90,0])
union(){
translate([0,0,-16])
cylinder (d=4,h=34);
translate([0,0,18])
sphere(5);
}}
```

Ce premier module appelé **rayon()** modélise un des rayons de la barre à roue

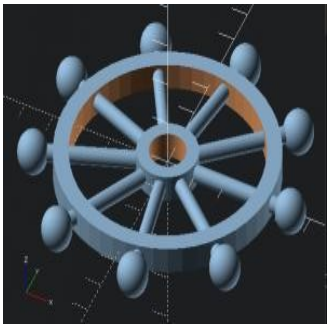


```
module roue(){
difference(){
cylinder(d=73,h=6,center=true);

cylinder(d=63,h=7,center=true);
}
difference(){
cylinder(d=20,h=6,center=true);

cylinder(d=12,h=7,center=true);
}}
```

Ensuite le module **roue()** modélise les éléments circulaires de la roue, deux éléments , le cercle extérieur et l'élément central sont crées par deux cylindres perforées grâce à l'instruction **difference(),**



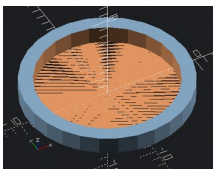
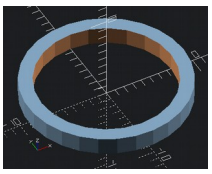
```
module barre(){
rayons = 9;
for(i = [1 : rayons])
{
rotate([0, 0, i * 360/rayons])
{
translate([25,0,0])
rayons();
}
roue();
}

barre();
```

La barre à roue est finalement assemblé dans le module **barre()** où le nombre de 9 rayons sont intégrés à la roue par l'instruction **for(i = [1 : n])**.

Le résultat sera affiché par l'instruction barre();.

Notre premier objet 3D, une bague

Nous allons faire une bague et pour ce faire, nous allons introduire la notion de **"module"**.

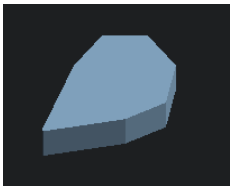
Un module est un court script que réalise une action qui peut être utilisée ultérieurement tout au long du programme.
On ne peut prévisualiser le résultat que si l'on appelle le module la fin du script (ici anneau();

```
$fn = 80;
module anneau(){
dia = 15;
difference(){
cylinder
(d=dia+2,h=2);
translate([0,0,-1])

cylinder
(d=15,h=4);
}}
```

Pour commencer nous allons créer le module **anneau()** où nous trouvons un cylindre qui traverse un autre et le évide grâce à la fonction **difference().**
Nous constatons pourtant que le cercle n'est pas entièrement percé, car le cercle traversant et appuyé sur le même niveau que le cercle extérieur. Pour éviter ceci il suffit de descendre l'axe **Z** d'un millimètre avec la commande **translate**.

Nous introduisons ici la notion de **variable**, qui est un mot ou une suite de caractères contenant une valeur ce qui permet éventuellement de appliquer des calculs à certaines valeurs (ici **$fn** et **dia**).




La variable **$fn = 60** augmente le nombre de facettes des cycindres.

```
$fn = 60
module goutte(){
hull(){
cylinder
(d=.1,h=2);
translate([-4,0,0])

cylinder (d=4,h=2);
}}
```

Une forme de goutte est crée en englobant **hull()** deux cercles , l'un d'un diamètre de 0.1 mm et l'autre de 3 mm.
C'est le deuxième module du programme : **goutte()**.

```
module bague(){
translate([0,1,0])
rotate([90,0,0])
anneau();
translate([3,0,7.5]
)
union(){
rotate([0,0,20])
goutte();
rotate([0,0,-20])
goutte();
}}
```

Le module **bague()** contient tous les éléments pour l'assemblage de la bague.
La commande **union()** sert à grouper tus les éléments qui seront affectés par l'instruction **translate** qui les précède et qui mettra tous les éléments à leur bonne place.

Pour visualiser le résultat de tout le programme appelez : **bague();**