

Conception : Introduction à Openscad, logiciel de conception 3D par codage.



Ateliers impression 3D open-source

par X. HINAULT

www.mon-club-elec.fr



Tous droits réservés - 2014.

Ce document légèrement payant est soumis au droit d'auteur et est réservé à l'usage personnel.

Afin d'encourager la production de supports didactiques de qualité, ce document est légèrement payant.

La licence d'utilisation est attribuée pour un usage personnel uniquement, dans le cercle familial. Mise en ligne et diffusion non autorisées.

Si vous n'êtes pas le détenteur de la licence attribuée pour l'usage de ce document, soyez sympa, merci d'acheter votre exemplaire personnel ici :

http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.ATELIERSIMPRESSION3D

Pour tout problème lié à l'utilisation de ce document, veuillez envoyer une copie ici : support@mon-club-elec.fr

Pour obtenir tout autres types de licence d'utilisation (enseignement, commercial, etc...), veuillez contacter l'auteur ici : support@mon-club-elec.fr

Vous avez constaté une erreur ? une coquille ? N'hésitez pas à nous le signaler à cette adresse : support@mon-club-elec.fr

Truc d'utilisation : visualiser ce document en mode diaporama dans le visionneur PDF. Navigation avec les flèches HAUT / BAS ou la souris.

En mode fenêtre, activer le panneau latéral vous facilitera la navigation dans le document. Bonne lecture !

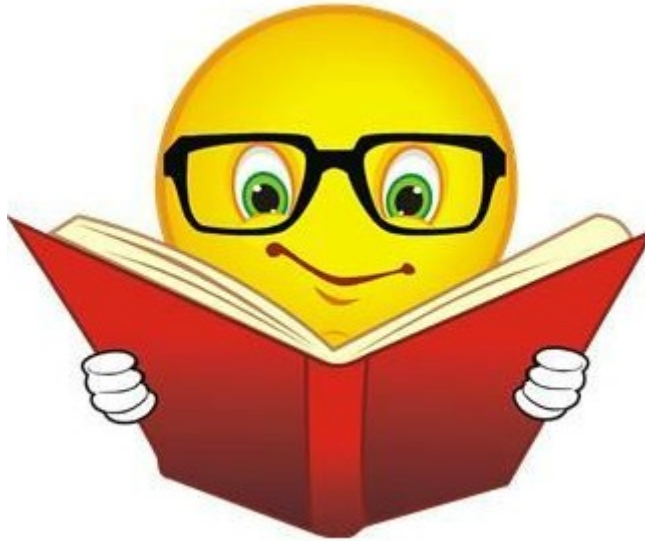
Licence de cet exemplaire accordée à Franck Ourion uniquement pour usage personnel, franck.ourion@univ-lorraine.fr # 7517226

Ateliers impression 3D open-source : Conception : Introduction à Openscad, logiciel de conception 3D par codage p. 1/30

1. Intro

L'objectif ici est :

- de découvrir Openscad, logiciel de conception par codage.

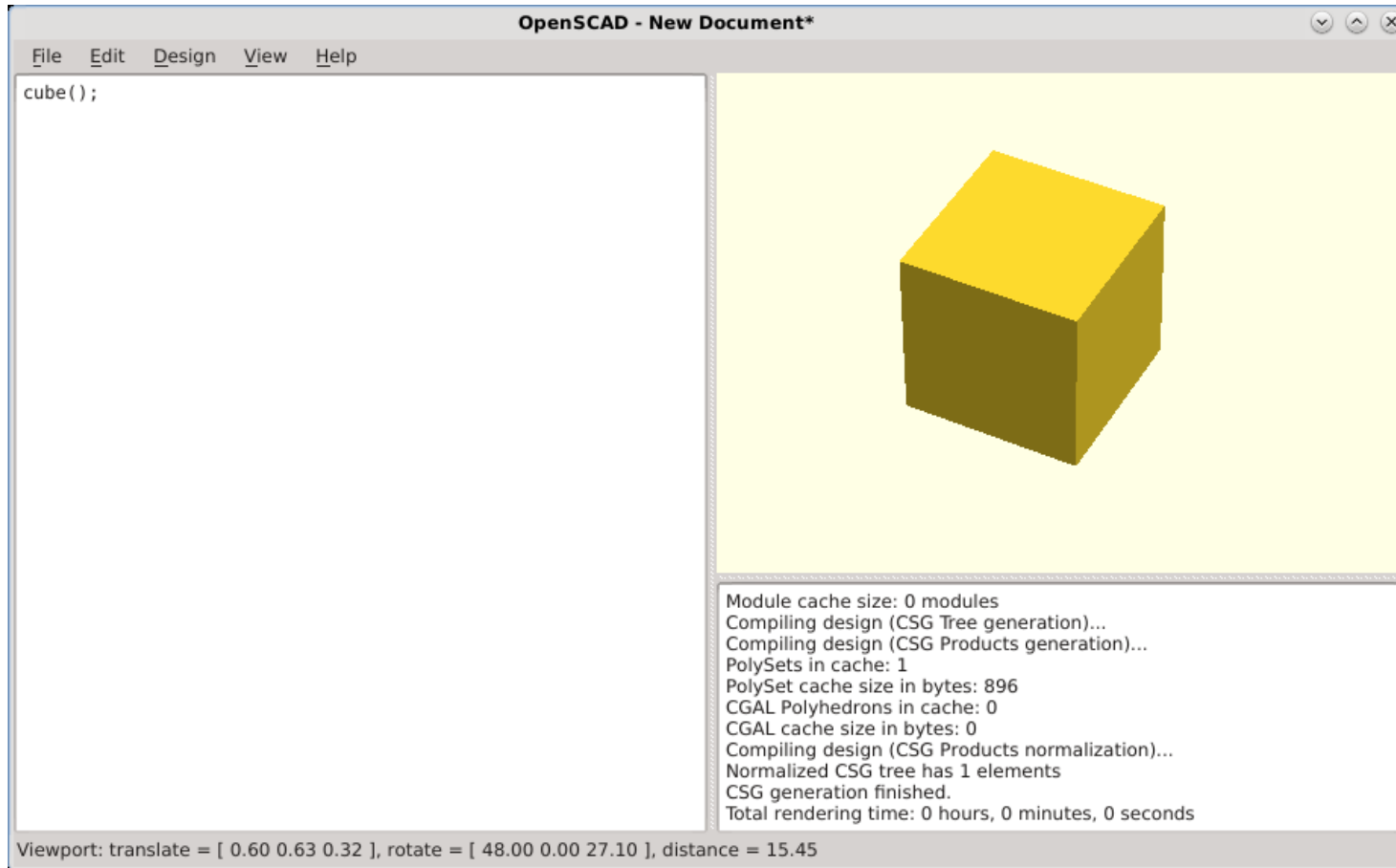


Prêt ? C'est parti !

2. Présentation

Openscad est un logiciel graphique de conception 3D par codage : il dispose logiquement :

- d'un éditeur de « code »
- d'une fenêtre de visualisation
- d'une console de sortie



3. Avantages/inconvénients

Avantages

Les avantages de cette façon de faire sont :

- rapidité de mise en œuvre pour des pièces simples
- réutilisation simple de pièces ou parties de pièces existantes par simple copier/coller
- possibilités de créer des pièces adaptées par simple modification de paramètres
- nombreux fichiers existants ré-utilisables
- courbe d'apprentissage assez rapide
- adapté pour des pièces symétriques, géométriques, avec éléments redondants

Inconvénients

Les inconvénients d'Openscad sont :

- prise en main moins intuitive qu'un logiciel graphique avec nécessité de se représenter mentalement la pièce 3D
- pas adapté pour des pièces complexes ou peu symétriques

4. Installation

Sous Ubuntu 12.04 LTS

On installe le dépôt suivant

```
$ sudo add-apt-repository ppa:chrysn/openscad
```

```
$ sudo apt-get update
```

```
$ sudo apt-get install openscad
```

Sous Debian Testing (SolydXK) :

Le logiciel est déjà dans les dépôts :

```
sudo apt-get install openscad
```

5. Lancement

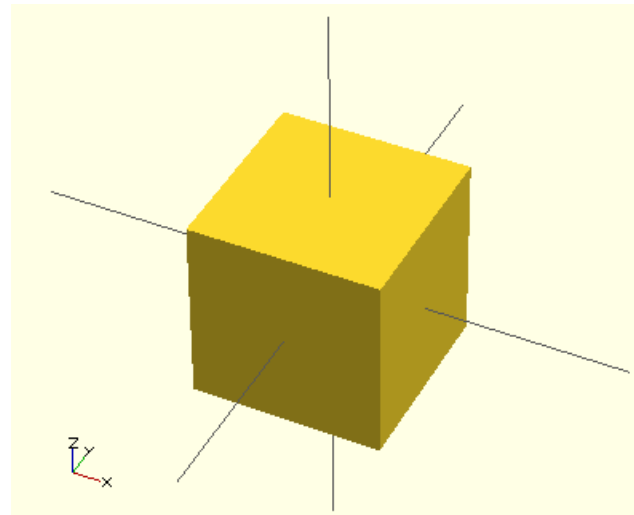
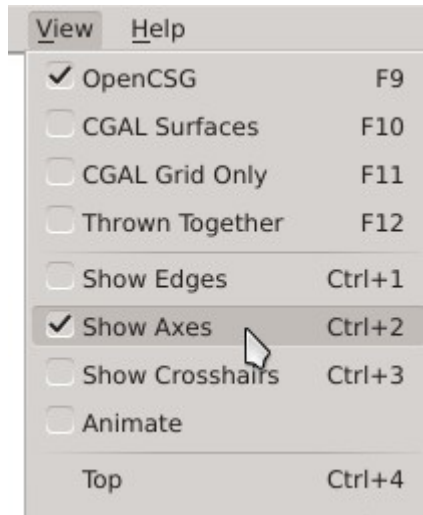
Menu graphique : Infographie > Openscad



Licence de cet exemplaire accordée à Franck Ourion uniquement pour usage personnel, franck.ourion@univ-lorraine.fr # 7517226

6. Utilisation de l'interface : commandes utiles

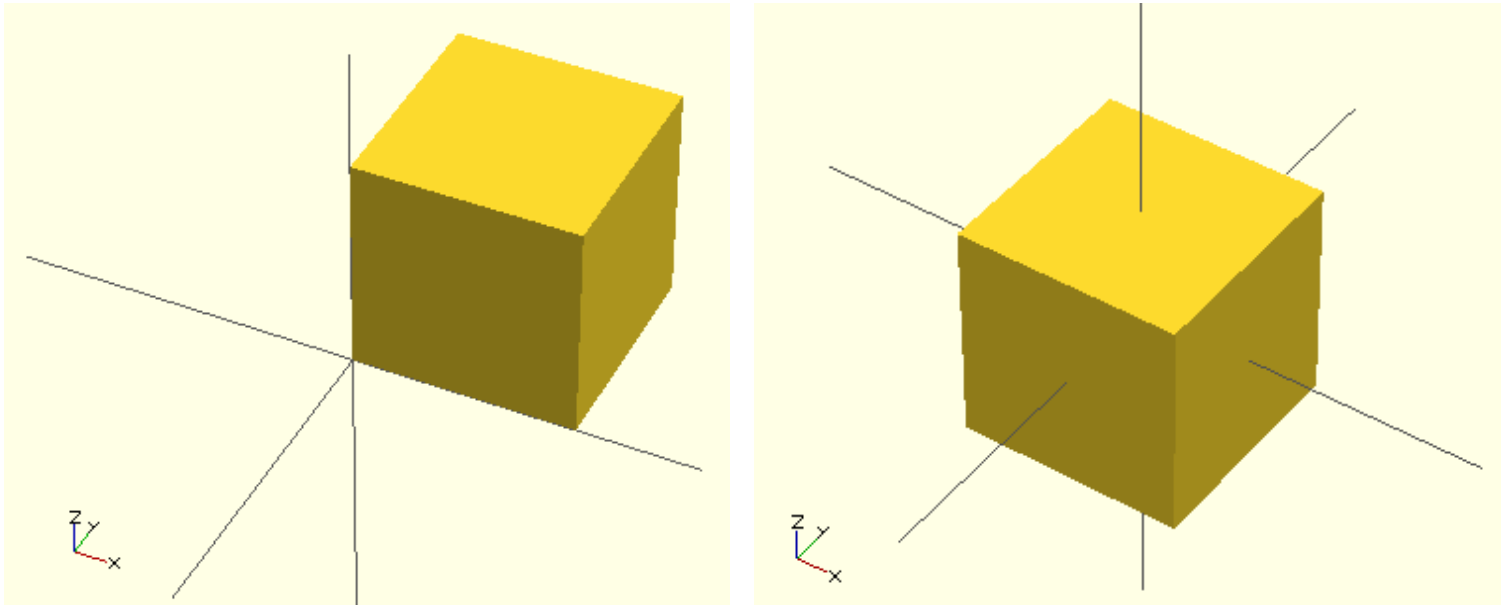
- Avec la souris :
 - la mollette : change la distance
 - clic gauche enfoncé : fait tourner le repère par rapport au centre de la vue
 - clic droit enfoncé : déplace le repère dans la fenêtre
- Pour afficher les axes : view > show axe
 - par défaut, le centre est au centre



7. Le système de coordonnées

On a 2 possibilités pour l'utilisation du système de coordonnées :

- soit le mode par défaut, « non centré » :
- soit le mode centré que l'on précisera avec une option de la forme `center=true` : le (0,0,0) sera au centre de la pièce. C'est ce mode qui offre le plus de souplesse pour intégrer une première pièce dans une autre.



à gauche mode par défaut « non centré », à droite mode «centré » (`center=true`)

8. Principe général d'utilisation

Ecrire le code de description de la pièce

La pièce 3D va être générée à l'aide d'un code de programmation. Ce code est de type C, c'est à dire respectant la syntaxe suivante :

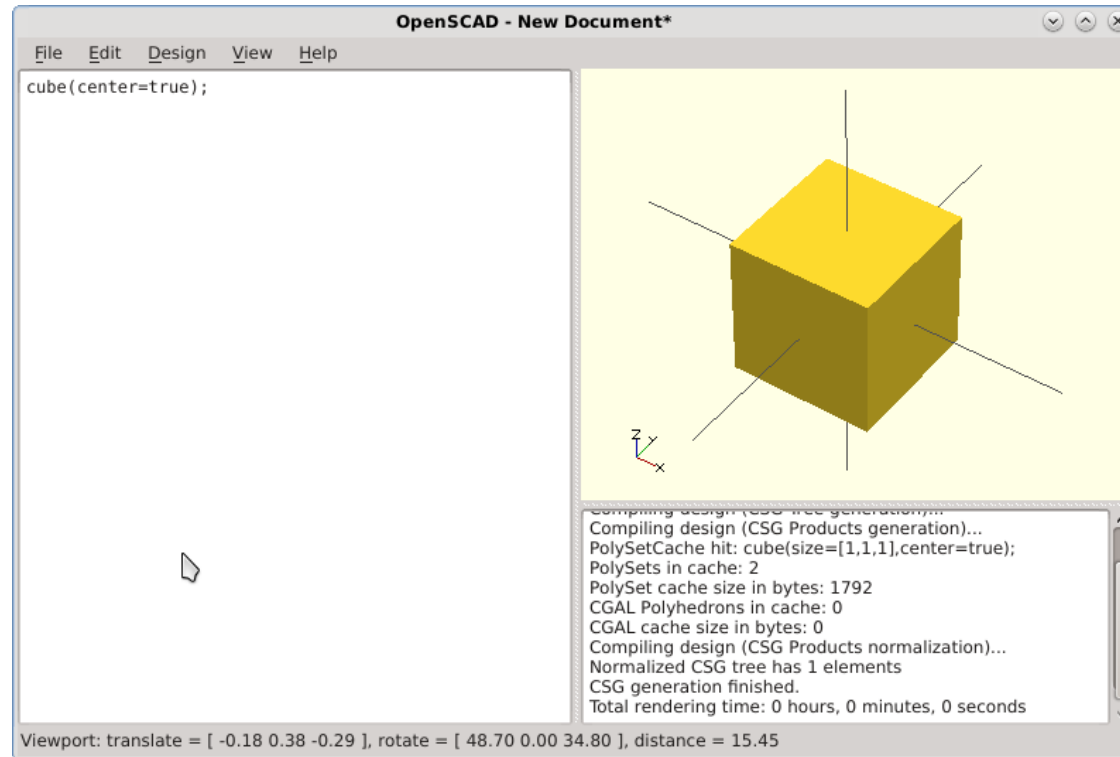
- ; de fin de ligne
- { } de début et fin de section de code
- supportant la boucle for

Les primitives 3D (cube, cylindre, etc...) sont générées à l'aide d'une simple instruction. Par exemple, pour générer un cube, on fera :

```
cube(center=true);
```

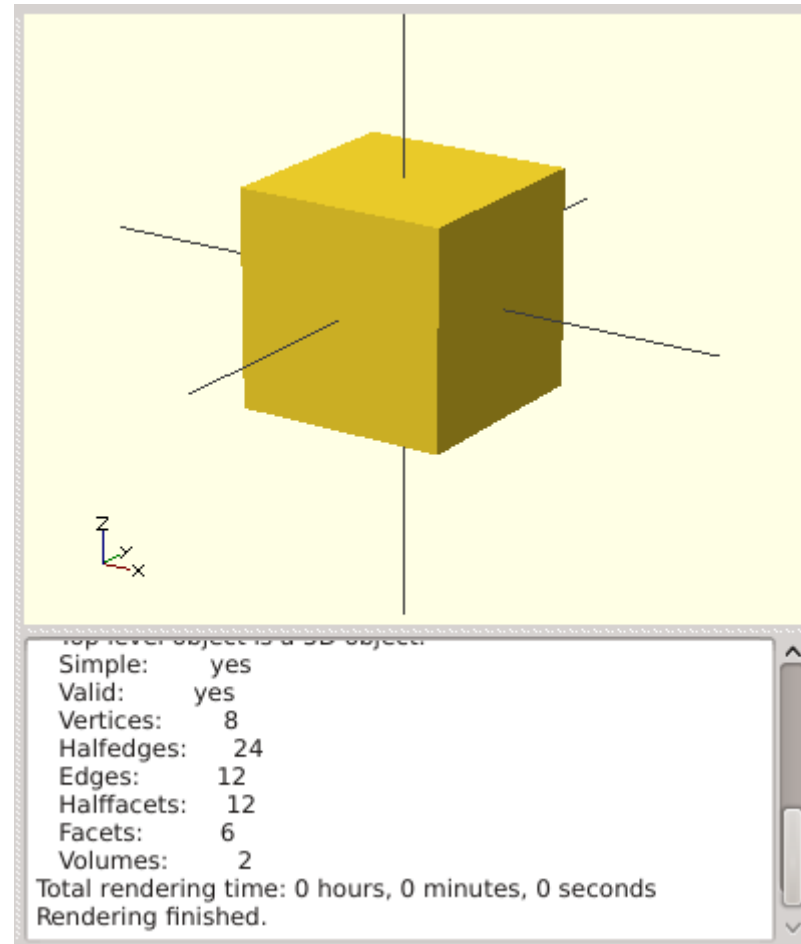
Visualiser la pièce en mode « allégé »

Une fois le code écrit, on affiche la pièce en utilisant la commande Design> Compile ou <F5> ce qui donne :



Générer la pièce 3D détaillée

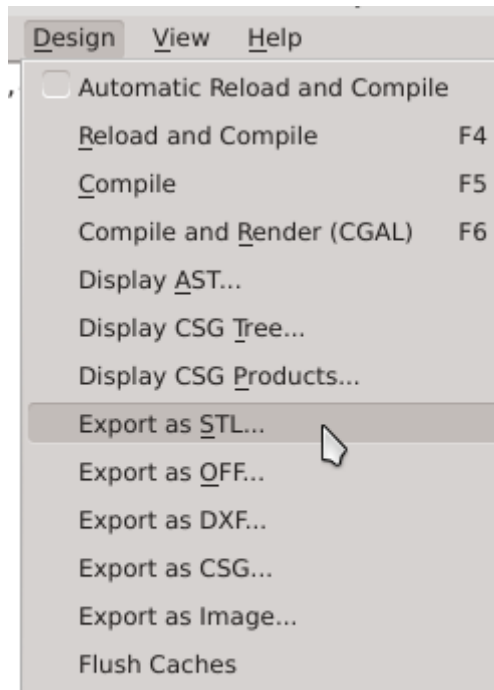
La vue obtenue avec la commande <compile> est une vue simplifiée : pour pouvoir générer un fichier *.STL, le seul utile en impression 3D, il faut générer la pièce 3D détaillée. C'est plus long, donc le faire seulement quand la pièce est finie. On fait ça avec le menu : Design > compile and Render ou <F6>



En apparence, c'est à peu près semblable au résultat précédent, mais ici le rendu est complet.

Exporter au format *.STL

Une fois la pièce détaillée obtenue, on peut l'exporter au format *.STL via le menu : Design > export as STL



Impression de la pièce 3D

L'impression se fait ensuite en générant le G-Code à l'aide de Slic3R, comme on le ferait pour n'importe quel autre fichier *.STL.

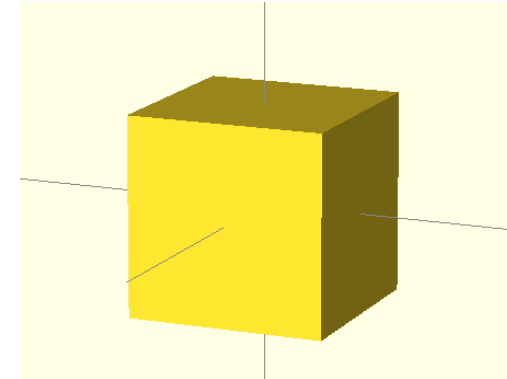
Une fois le fichier *.gcode obtenu, on peut lancer l'impression de la pièce.

9. Les primitives 3D de base : Cube

Syntaxe : http://en.wikibooks.org/wiki/OpenSCAD_User_Manual/The_OpenSCAD_Language#Primitive_Solids

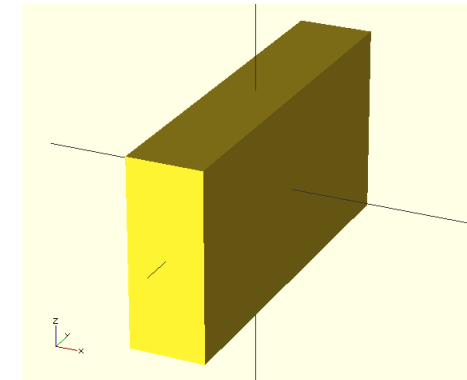
Un cube simple :

```
cube(10,true); // cube 10 centré en 000
```



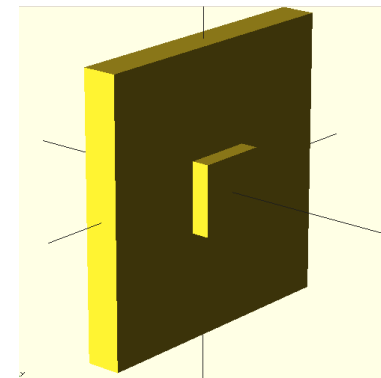
Un parallépipède :

```
cube([10,50,25],true); // parallépipède centré en 000
```



Cube encastré dans un parallépipède

```
cube ([5,40,40],true); // parap centré en O  
cube (10,true); // cube centré en O
```

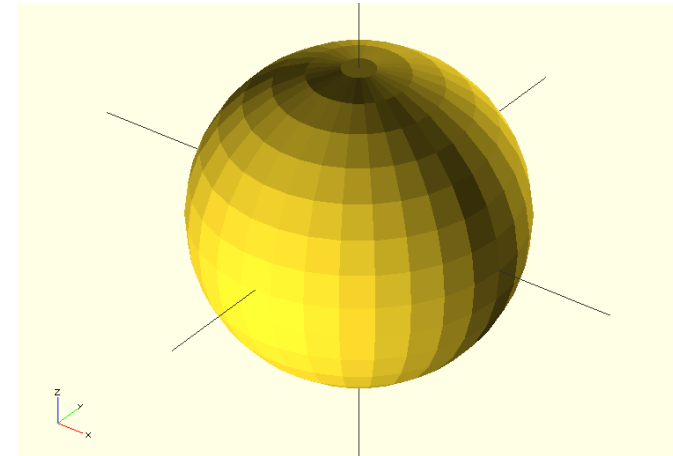


10. Les primitives 3D de base : Sphere

Syntaxe : http://en.wikibooks.org/wiki/OpenSCAD_User_Manual/The_OpenSCAD_Language#Primitive_Solids

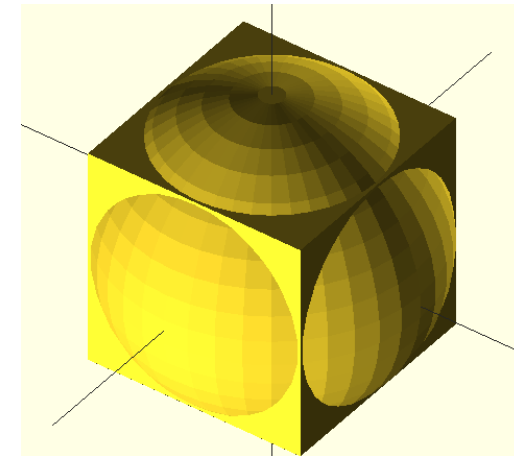
Une sphère simple de résolution moyenne et rayon 2

```
sphere(2, $fn=30);
```



Une sphère encastrée dans un cube

```
cube(10,true);  
sphere(7, $fn=40);
```

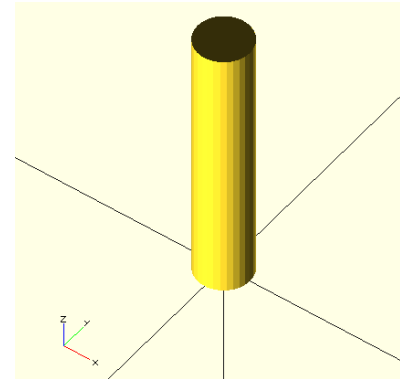


11. Les primitives 3D de base : *Cylinder*

Syntaxe : http://en.wikibooks.org/wiki/OpenSCAD_User_Manual/The_OpenSCAD_Language#Primitive_Solids

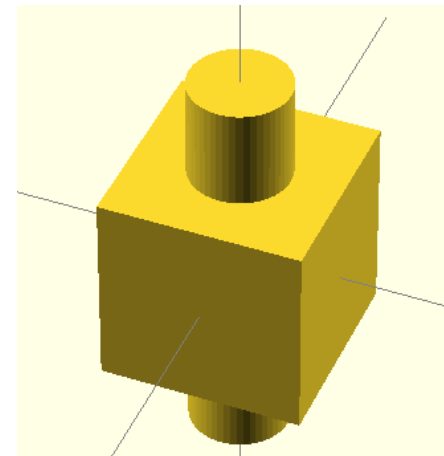
Un cylindre simple

```
cylinder(h = 50, r=5, $fs=1);
```



Un cylindre traversant un cube :

```
cube(10, true);  
cylinder(h=20, r=2.5, $fn=50, center=true);
```



Remarques :

- Attention : diamètre = 2 x rayon !!
- Pour réaliser un pas de vis M3, on pourra utiliser un cylindre de 2,75mm soit $r=1,37$

12. Les opérations de positionnement *sur les primitives 3D de base*

Documentation : http://en.wikibooks.org/wiki/OpenSCAD_User_Manual/Transformations

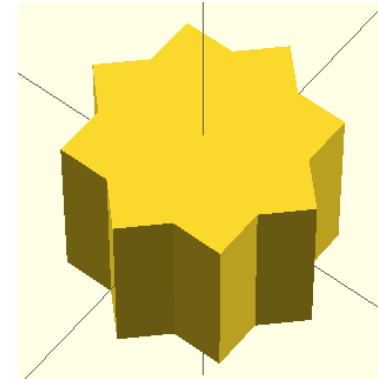
Les plus importantes à maîtriser ++

On utilise en paramètre pour ces fonctions un ensemble de 3 valeurs sous la forme [X,Y,Z]

rotate

Redimensionne un élément

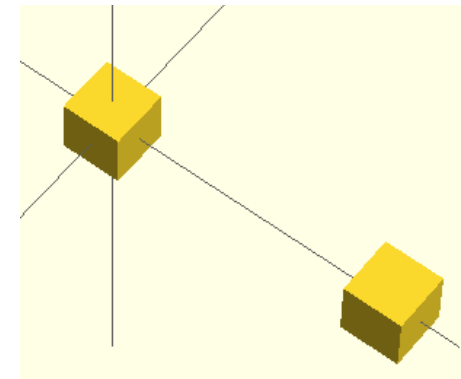
```
cube(10, center=true);  
rotate ([0,0,45]) cube(10,center=true);
```



translate

Redimensionne un élément

```
cube(10, center=true);  
translate ([50,0,0]) cube(10,center=true);
```



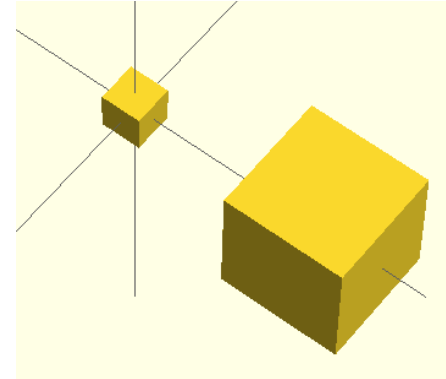
13. Les opérations de taille *sur* les primitives 3D de base

Documentation : http://en.wikibooks.org/wiki/OpenSCAD_User_Manual/Transformations

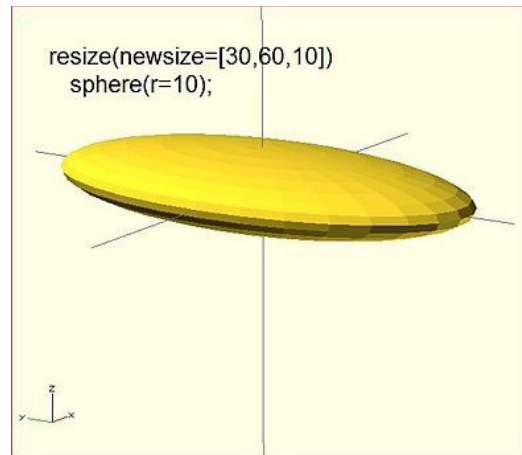
scale

Redimensionne un élément

```
cube(10, center=true);  
translate ([50,0,0]) scale(3) cube(10,center=true);
```



resize



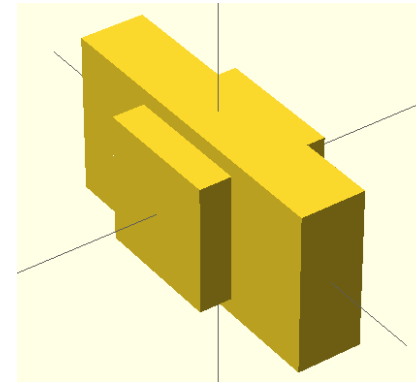
14. Les opérations *entre primitives 3D*

Documentation : http://en.wikibooks.org/wiki/OpenSCAD_User_Manual/CSG_Modelling

Union

Permet de réunir plusieurs éléments en 1 seul : ceci permet d'appliquer ensuite une opération sur l'ensemble des éléments de l'union d'un coup.

```
union() {  
    cube(20,true); // cube 20 centré en 000  
    cube([10,50,25],true); // parallépipède centré en 000  
} // fin union
```

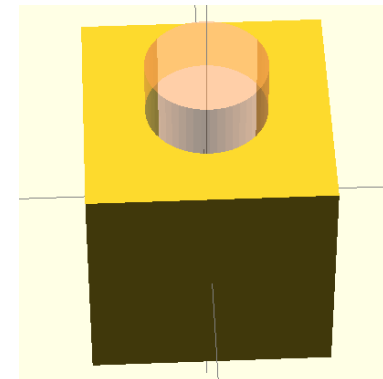


Soustraction

Permet de soustraire une primitive d'une autre : une opération typiquement utilisée pour un trou, etc...

Truc : pour visualiser l'élément soustrait, mettre un # devant

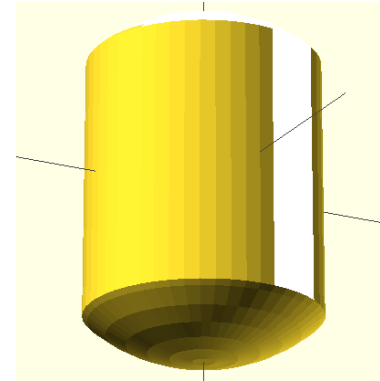
```
difference() {  
    cube(20,true); // cube 10 centré en 000  
    #cylinder(h=30,r=5,$fn=50,center=true); // cylindre en 000  
} // fin difference
```



Intersection

Permet d'obtenir l'intersection entre 2 primitives

```
intersection() {  
    sphere(15,$fn=50, true); // sphere centrée en 000  
    cylinder(h=30,r=10,$fn=50,center=true); // cylindre en 000  
} // fin intersection
```



15. Enchaînement d'opérations sur primitives : syntaxes possibles

Deux syntaxes possibles :

- soit façon fonction()

```
difference() {  
    cube(20,true); // cube 10 centré en 000  
    #cylinder(h=30,r=5,$fn=50,center=true); // cylindre en 000  
} // fin difference
```

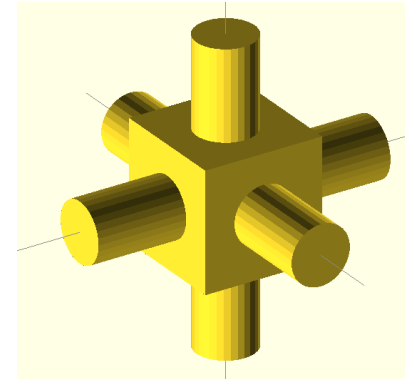
- soit sur une ligne :

```
translate ([50,0,0]) scale(3) cube(10,center=true);
```

16. Quelques exemples de combinaisons simples

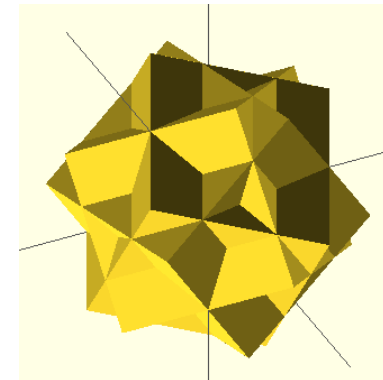
Trois cylindres encastrés dans un cube

```
cube(20,true);  
  
translate([0,0,-25]) {  
    cylinder(h = 50, r=5, $fs=1);  
}  
  
translate([-25,0,0]) {  
    rotate ([0,90,0]) cylinder(h = 50, r=5, $fs=1);  
}  
  
rotate([90,0,0]) {  
    translate ([0,0,-25]) cylinder(h = 50, r=5, $fs=1);  
}
```



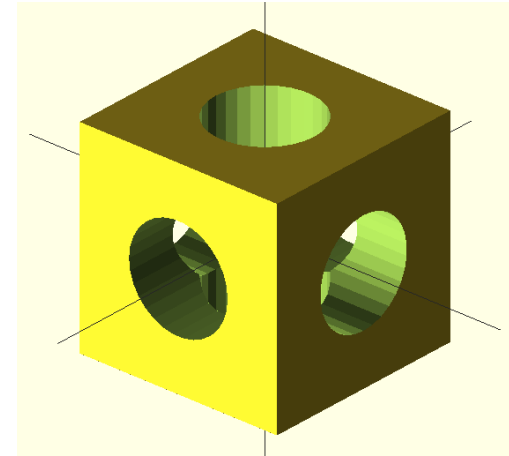
Rotation de 3 cubes identiques, chacun de 45° autour d'un axe

```
rotate(a=[45,0,0]) {  
    cube (10,true);  
}  
  
rotate(a=[0,45,0]) {  
    cube (10,true);  
}  
  
rotate(a=[0,0,45]) {  
    cube (10,true);  
}
```



Soustraction de 3 cylindres à 1 cube

```
Difference () {  
cube(20,true); // premier élément  
  
translate([0,0,-25]) {  
    cylinder(h = 50, r=5,$fs=1);  
}  
  
translate([-25,0,0]) {  
    rotate ([0,90,0]) cylinder(h = 50, r=5, $fs=1);  
}  
  
rotate([90,0,0]) {  
    translate ([0,0,-25]) cylinder(h = 50, r=5, $fs=1);  
}  
} // fin difference
```



17. Prendre de bonnes habitudes : utilisation de variables

Il est possible, et même très conseillé d'utiliser des variables : de cette façon, il sera possible d'adapter la pièce très simplement en changeant uniquement la valeur des variables.

```
// variables
haut=10;
long=50;
larg=20;

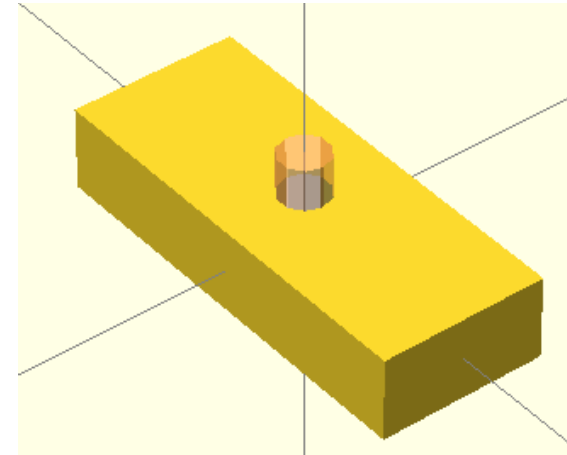
diam=6;

// la pièce
difference(){

    cube([long,larg,haut], center=true);

    #cylinder(h=haut*2, r=diam/2, center=true);

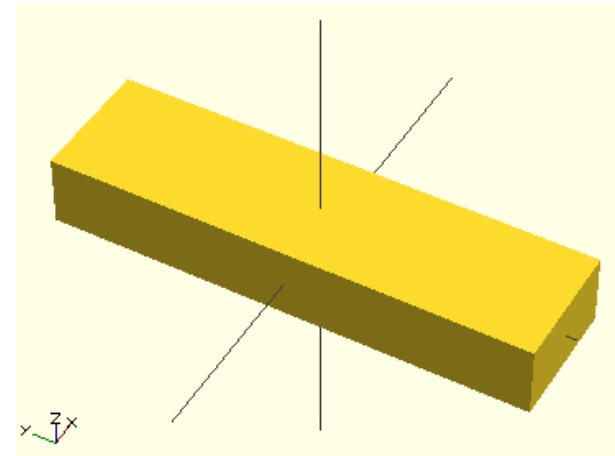
} // difference
```



18. Un premier exemple de pièce simple

Pour commencer, je vous propose de créer une pièce très simple : un bloqueur de Y Idler (la pièce qui appuie sur le filament) sur la Prusa i3 qui permet de changer facilement le filament de l'imprimante. Il s'agit d'un simple parallélépipède de 8mmx30mmx4,2mm ce qui donne :

```
cube([8,30,4.2],center=true);
```



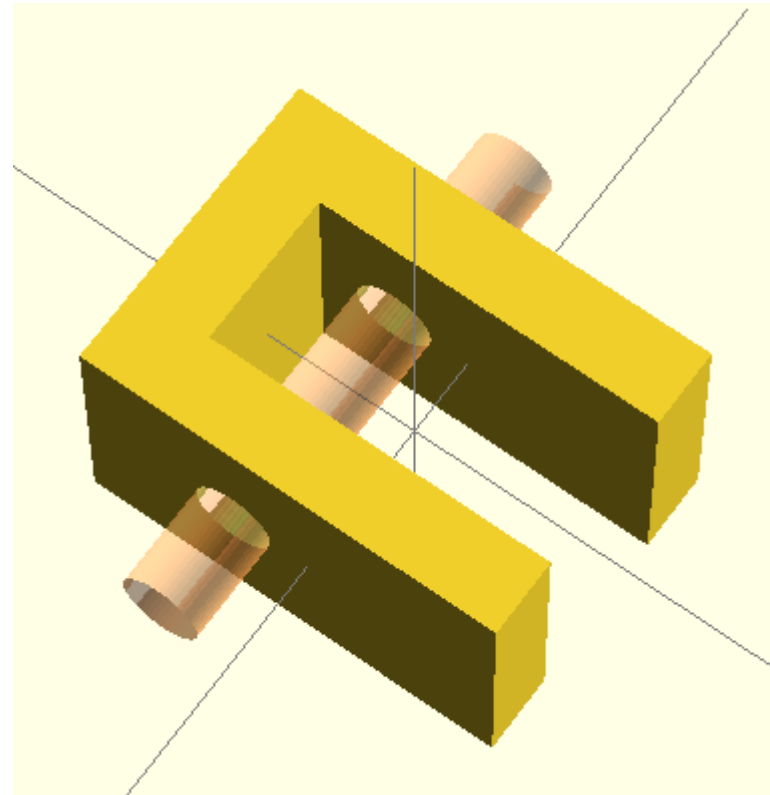
Très simple... mais très très pratique !

19. Un autre exemple de pièce : un clip pour panneau 6mm avec axe de serrage

```
ep=3;
difference(){
  union(){
    rotate([0,90,0]) {
      translate ([0,0,3+ep/2]) cube([10,15,ep],center=true);
      translate ([0,0,-(3+ep/2)]) cube([10,15,ep],center=true);
    }

    rotate([90,90,0])translate ([0,0,15/2]) cube([10,6+2*ep,ep],center=true);
  } // fin union

  #rotate([0,90,0])translate([0,-3,0])cylinder(h=20, r=1.5, $fn=50, center=true);
} // fin difference
```



20. Un autre exemple de pièce : un support pour pince à épiler

La pince à épiler sert à « attraper » le surplus de filament qui sort de la buse de l'imprimante 3D : la pince à épiler, on la cherche tout le temps... Un petit support fixé sur le cadre de la Prusa... et hop, on ne cherche plus la pince à épiler.. Pratique !

Une extension du clip simple :

```
//--- le clip ---
ep=3;

difference(){
union(){
rotate([0,90,0]) {

    translate ([0,0,3+ep/2]) cube([10,15,ep],center=true);
    translate ([0,0,-(3+ep/2)]) cube([10,15,ep],center=true);

}

    rotate([90,90,0])translate ([0,0,15/2]) cube([10,6+2*ep,ep],center=true);

} // fin union

    #rotate([0,90,0])translate([0,-3,0])cylinder(h=20, r=1.5, $fn=50, center=true);

} // fin différence

//--- support de la pince
translate ([0,-15/2-10/2-ep/2,-10/2+ep/2]){

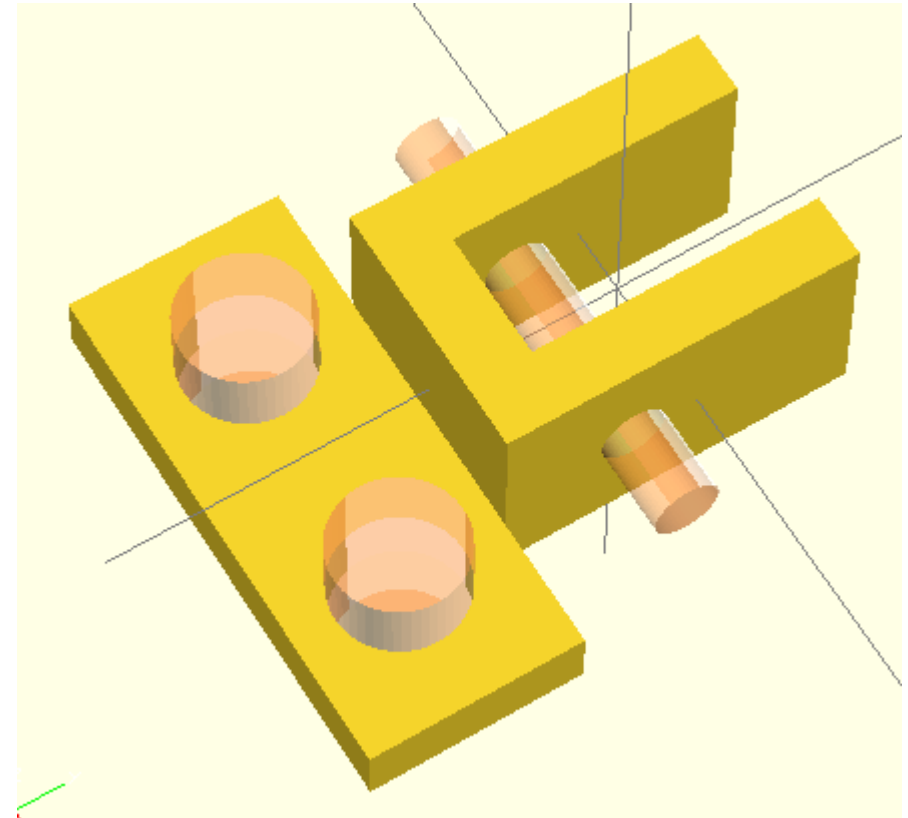
    difference(){

        cube([24,10,ep],center=true);

        #translate([-6,0,0])cylinder(h=10, r=3, $fn=50, center=true);
        #translate([6,0,0])cylinder(h=10, r=3, $fn=50, center=true);

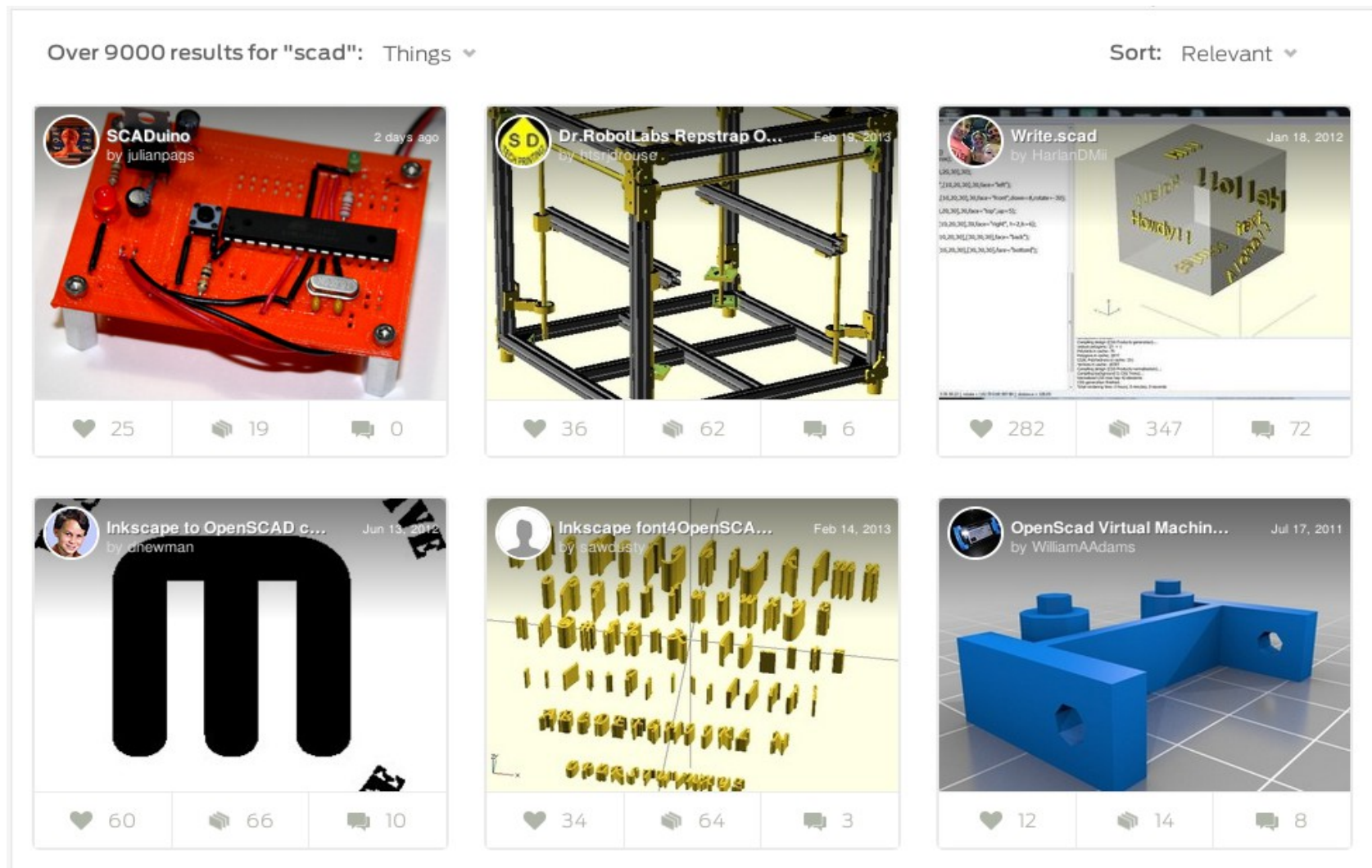
    } // fin difference

} // fin translate
```



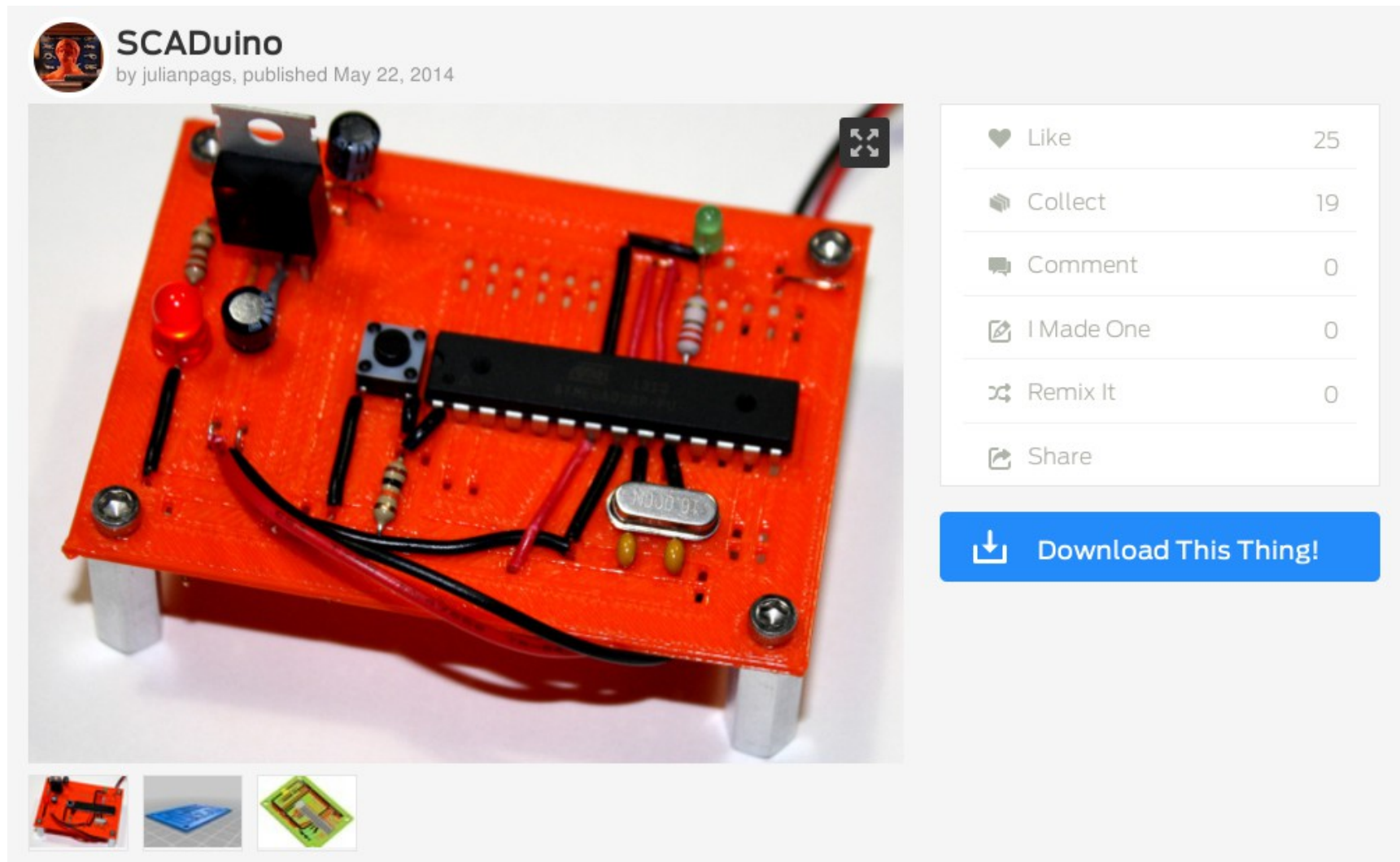
21. Des fichiers openscad « clés en main »

Des tas sur thingiverse en cherchant avec le mot clé « scad » : de très nombreuses pièces facilement adaptables sans besoin de logiciel complexe !



22. Tiens, tiens... intéressant !

<http://www.thingiverse.com/thing:338744>



23. Quelques trucs utiles

Garder visible les primitives extrudées

Ajouter un # devant la primitive

24. Usages avancés

Avec Openscad, il est possible :

- de créer des modules
- d'utiliser une pièce dans une autre pièce
- d'utiliser un SVG convertit en *.scad par un plugin inkscape : voir <http://www.thingiverse.com/thing:24808>

25. A présent, vous devriez être capable :

- D'utiliser Openscad pour créer des pièces simples.

Table des matières

Conception : Introduction à Openscad, logiciel de conception 3D par codage.

- Intro |
- Présentation |
- Avantages/inconvénients |
- Installation |
- Lancement |
- Utilisation de l'interface : commandes utiles |
- Le système de coordonnées |
- Principe général d'utilisation |
- Les primitives 3D de base : Cube |
- Les primitives 3D de base : Sphere |
- Les primitives 3D de base : Cylinder |
- Les opérations de positionnement sur les primitives 3D de base |
- Les opérations de taille sur les primitives 3D de base |
- Les opérations entre primitives 3D |
- Enchaînement d'opérations sur primitives : syntaxes possibles |
- Quelques exemples de combinaisons simples |
- Prendre de bonnes habitudes : utilisation de variables |
- Un premier exemple de pièce simple |
- Un autre exemple de pièce : un clip pour panneau 6mm avec axe de serrage |
- Un autre exemple de pièce : un support pour pince à épiler |
- Des fichiers openscad « clés en main » |
- Tiens, tiens... intéressant ! |
- Quelques trucs utiles |
- Usages avancés |
- A présent, vous devriez être capable : |

Bravo !
vous avez terminé cet atelier !



Prêt pour la suite ? Retrouvez de nombreux autres thèmes d'ateliers « Impression 3D » ici :
http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.ATELIERSIMPRESSIION3D