Getting Started with OpenSCAD

"Geometric 3D Modeling for Programmers"

Introduction

This handout was provided to attendees at a 30 minute introduction to OpenSCAD at the OMSI Mini Maker Faire in Portland, Oregon on Saturday and Sunday, September 15 and 16, 2018. This is barely sufficient time to get your feet wet and whet your appetite. I welcome questions and comments via email.

This handout is available in PDF format at http://www.herbweiner.com/gsOpenSCAD.pdf (or scan the QR code below). All of the links in the PDF are clickable, so you can easily access all of the examples and references.

Installation

OpenSCAD is free software for Windows, Macintosh, and Linux.

Install OpenSCAD version 2019.05 or newer: http://www.openscad.org/downloads.html.

To enable the Customizer: *Edit* > *Preferences* > *Features* > *Enable Customizer*.

You **may** need to uncheck *Hide editor*, *Hide console*, and *Hide Customizer* in the *View* menu.

References

See also the links for the five examples in the next column.

- https://en.wikibooks.org/wiki/OpenSCAD_User_Manual, User Manual.
- http://www.openscad.org/cheatsheet/index.html, Cheat Sheet (see other side).
- https://all3dp.com/2/how-to-do-parametric-3d-modeling-with-free-software, Herb Weiner, How to Do Parametric 3D Modeling with Free Software.
- https://all3dp.com/2/openscad-tutorial-for-beginners-5-easy-steps, OpenSCAD Tutorial for Beginners (5 Easy Steps).
- https://cubehero.com/2013/11/19/know-only-10-things-to-be-dangerous-in-openscad, Know only 10 things to be dangerous in OpenSCAD.
- https://www.thingiverse.com/thing:2812634, How to run Customizer on your own computer.
- http://customizer.makerbot.com/docs, Developer Documentation for Customizer.
- https://www.amazon.com/gp/product/1484213246, *3D Printed Science Projects: Ideas for your classroom, science fair or home (Technology in Action)*, Joan Horvath and Rich Cameron, ISBN 978-1-4842-1324-7.

http://www.herbweiner.com/gsOpenSCAD.pdf
2019-08-22

Herb Weiner <herbw@wiskit.com>

OpenSCAD User Interface

Each OpenSCAD window is divided into three or four panes. The leftmost pane is the Text Editor pane, which contains the OpenSCAD code. To the right of the Text Editor pane are the Viewing Area pane above and a Console pane below. In some cases, the Customizer pane may be present to the right of the Viewing Area and Console panes.

There are Toolbars at the top of the Text Editor pane and at the bottom of the Viewing Area pane. Hovering the cursor over any tool in either Toolbar displays the name of the tool.

Examples used for the Presentation

Many of these examples include code that is commented out (using // at the beginning of a line, or using /* and */ within a line). By uncommenting some of this text, we can observe how various changes affect the OpenSCAD model.

- http://www.herbweiner.com/example1.scad

  This example introduces the three basic shapes (sphere, cube, and cylinder or cone) and two important transformations (translation and rotation).

- http://www.herbweiner.com/example2.scad

  This example explores how more compex shapes can be constructed from basic shapes using the union, intersection, and difference operations.

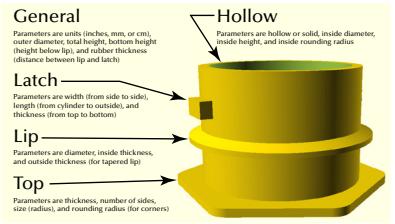
- http://www.herbweiner.com/example3.scad

  This example explores how parameters can be used to customize models.

- http://www.herbweiner.com/example4.scad

  This example explores modules can be used to construct more complex models.

- https://www.thingiverse.com/thing:2976380

  This Blender Top Filler Cap from Thingiverse demonstrates a more complex, customizable model. This model contains about 20 parameters in five groups, and utilizes all of the concepts introduced by the earlier examples. The intent of this example is not to explain every line of code, but rather to illustrate how a complex model can be constructed using basic shapes, transformations, constructive solid geometry operations, and parameters, and how an OpenSCAD program can be structured to make it easier to understand.

OpenSCAD Caveats / Gotchas / Cautions / Pitfalls

- The default units in OpenSCAD are millimeters. To work in inches, scale the entire model by 25.4.
- In most cases (with a few exceptions like module and function parameters and for loop variables), the value of variables is determined at compile time and can not be changed at run time; thus, variables behave more like constants.
- The difference operation can leave infinitesimally thin surfaces, as shown in example 4. To avoid this problem, ensure that surfaces do not precisely overlap.

Syntax

var = value;

module name(…) { … }
name();

function name(…) = …
name();

include <….scad>

use <….scad>

2D

circle(radius | d=diameter)

square(size,center)

square([width,height],center)

polygon([points])

polygon([points],[paths])

text(text, size, font,
halign, valign, spacing,
direction, language, script)

3D

sphere(radius | d=diameter)

cube(size, center)

cube([width,depth,height], center)

cylinder(h,r|d,center)

cylinder(h,r1|d1,r2|d2,center)

polyhedron(points, triangles, convexity)

Transformations

translate([x,y,z])

rotate([x,y,z])

scale([x,y,z])

resize([x,y,z],auto)

mirror([x,y,z])

multmatrix(m)

color("colorname",alpha)

color([r,g,b,a])

offset(r|delta,chamfer)

hull()

minkowski()

Boolean operations

union()

difference()

intersection()

Modifier Characters

*	disable
!	show only
#	highlight / debug
%	transparent / background

Mathematical

abs

sign

sin

cos

tan

acos

asin

atan

atan2

floor

round

ceil

ln

len

let

log

pow

sqrt

exp

rands

min

max

Functions

concat

lookup

str

chr

search

version

version_num

norm

cross

parent_module(idx)

Other

echo(…)

for (i = [start:end]) { … }

for (i = [start:step:end]) { … }

for (i = […,…,…]) { … }

intersection_for(i = [start:end]) { … }

intersection_for(i = [start:step:end]) { … }

intersection_for(i = […,…,…]) { … }

if (…) { … }

~~assign (…) { … }~~

import("….stl")

linear_extrude(height,center,convexity,twist,slices,scale)

rotate_extrude(angle,convexity)

surface(file = "….dat",center,convexity)

projection(cut)

render(convexity)

children([idx])

List Comprehensions

Generate [for (i = *range|list*) i]

Conditions [for (i = …) if (condition(i)) i]

Assignments [for (i = …) let (assignments) a]

Special variables

$fa	minimum angle
$fs	minimum size
$fn	number of fragments
$t	animation step
$vpr	viewport rotation angles in degrees
$vpt	viewport translation
$vpd	viewport camera distance
$children	number of module children

Links: Official website | Code | Issues | Manual | MCAD library | Forum | Other links