

OpenSCAD Help

Table of contents

Introduction	10
What's new	10
Getting Started	10
Language Reference	12
The OpenSCAD Language	12
Introduction	12
Comments	13
Values and Data Types	13
Numbers	13
Boolean Values	14
Strings	15
Ranges	15
The Undefined Value	16
Variables	16
Undefined variable	17
Scope of variables	17
Variables are set at compile-time, not at run-time	18
Special Variables	19
Vectors	19
Vector Operators (concat & len)	20
Matrix	21
Getting input	21
3D Objects	22
Primitive Solids	22
cube	22
sphere	22
cylinder	22
polyhedron	22
Debugging polyhedra	22
Mis-ordered faces	22
Point repetitions in a polyhedron point list	22
3D to 2D Projection	23
2D Objects	23
square	23
circle	23
ellipse	23
regular polygon	23
polygon	23
import_dxf	23
Text	23
Using Fonts & Styles	23
Alignment	23
Vertical Alignment	23
Horizontal Alignment	24
3D Text	24
3D to 2D Projection	24
2D to 3D Projection	24

Linear Extrude	24
Usage	24
Twist	24
Center	24
Mesh Refinement	24
Scale	24
Rotate Extrude	24
Parameters	24
Usage	24
Examples	25
Mesh Refinement	25
Extruding a Polygon	25
Description of extrude Parameters	25
Extrude parameters for all extrusion modes	25
Extrude parameters for linear extrusion only	25
Transforms	25
Basic concept	25
Advanced concept	25
scale	25
resize	25
rotate	25
Rotation rule Help	25
translate	26
mirror	26
Function signature:	26
Examples	26
multimatrix	26
More?	26
color	26
function signature	26
Example	26
Example 2	26
offset	26
minkowski	26
hull	27
Combining transformations	27
Boolean Combinations	27
Boolean overview	27
2D examples	27
3D examples	27
union	27
difference	27
difference with multiple children	27
intersection	27
render	27
Other Functions and Operators	27
Condition and Iterator functions	27
For loop	28
Intersection For loop	28
if statement	28

else if	28
Conditional ?	28
Recursive function calls	28
Assign Statement	28
Let Statement	28
Mathematical Operators	28
Scalar Arithmetical Operators	28
Relational Operators	28
Logical Operators	28
Conditional Operator	29
Vector-Number Operator	29
Vector Operators	29
Vector Dot-Product Operator	29
Matrix Multiplication	29
Mathematical Functions	29
Trigonometric Functions	29
cos	29
sin	29
tan	29
acos	29
asin	29
atan	29
atan2	30
Other Mathematical Functions	30
abs	30
ceil	30
concat	30
cross	30
exp	30
floor	30
in	30
len	30
let	30
log	30
lookup	30
max	31
min	31
norm	31
pow	31
rands	31
round	31
sign	31
sqrt	31
Infinities and NaNs	31
String Functions	31
str	31
chr	31
ord	31
Also see search()	32
List Comprehensions	32

Basic Syntax	32
Multiple generator expressions	32
for	32
each	32
if	32
if/else	32
let	32
Nested loops	32
Advanced Examples	32
Generating vertices for a polygon	32
Flattening a nested vector	32
Sorting a vector	33
Selecting elements of a vector	33
Concatenating two vectors	33
Other Language Features	33
Special Variables	33
\$fa, \$fs and \$fn	33
\$t	33
\$vp, \$vpt and \$vpd	33
\$preview	33
Echo Statements	33
Usage examples	33
Rounding examples	33
Small and large Numbers	33
HTML	34
Echo Function	34
Render	34
Surface	34
Text file format	34
Images	34
Examples	34
Search	34
Search Usage	34
Search Arguments	34
Search Usage Examples	34
Index values return as list	35
Search on different column; return index values	36
Search on list of values	37
Search on list of strings	38
Getting the right result	39
OpenSCAD Version	39
parent_module(n) and \$parent_modules	39
assert	39
Example	39
Checking parameters	39
Adding message	39
Using assertions in functions	39
User Defined Functions and Modules	39
Introduction	39
Scope	39

Functions	39
Recursive Functions	39
Function Literals	40
Modules	40
Object modules	40
Operator Modules	40
Children	40
Further Module Examples	40
Recursive Modules	40
Overwriting built-in modules	40
Overwriting built functions	40
Debugging Aids	40
Advanced Concept	40
Background Modifier	40
Debug Modifier	41
Root Modifier	41
Disable Modifier	41
Echo Statements	41
External Libraries and code files	41
Use and Include	41
Directory separators	41
Variables	41
Scope of Variables	41
Overwriting variables	41
Example "Ring-Library"	41
Nested Include and Use	41
Import	41
Parameters	42
Convexity	42
Notes	42
Import DXF	42
Import STL	42
surface	42
Parameters	42
Text file Format	42
Images	42
Examples	42
MCAD Library	42
regular_shapes.scad	44
2D regular shapes	44
regular_polygon(sides, radius)	44
n-gons 2D shapes	44
triangle(radius)	44
pentagon(radius)	44
hexagon(radius)	44
heptagon(radius)	45
octagon(radius)	45
nonagon(radius)	45
decagon(radius)	45
hendecagon(radius)	45

dodecagon(radius)	45
ring(inside_diameter, thickness)	45
ellipse(width, height)	45
egg_outline(width, thickness)	45
3D regular shapes	45
cone	45
oval_prism	45
oval_tube	46
cylinder_tube	46
triangle_prism	46
triangle_tube	46
pentagon_prism	46
pentagon_tube	46
hexagon_prism	46
hexagon_tube	46
heptagon_prism	46
heptagon_tube	46
octagon_prism	46
octagon_tube	46
nonagon_prism	46
decagon_prism	47
hendecagon_prism	47
dodecagon_prism	47
torus	47
torus2	47
oval_torus	47
triangle_pyramid	47
square_pyramid	47
egg	47
involute_gears.scad	47
bevel_gear_pair()	47
bevel_gear()	47
gear()	48
Tests	48
test_gears()	48
test_meshing_double_helix()	48
test_bevel_gear()	48
test_bevel_gear_pair()	48
test_backlash()	48
dotSCAD Library	48
2D modules	48
arc	48
pie	48
rounded_square	48
line2d	49
polyline2d	49
hull_polyline2d	49
hexagons	49
polytransversals	49
multi_line_text	49

voronoi2d	49
3D modules	49
rounded_cube	49
rounded_cylinder	49
crystal_ball	49
line3d	49
polyline3d	49
hull_polyline3d	50
function_grapher	50
sweep	50
loft	50
starburst	50
voronoi3d	50
Transformations	50
along_width	50
hollow_out	50
bend	50
shear	50
2D functions	50
in_shape	51
bijection_offset	51
trim_shape	51
triangulate	51
contours	51
2D/3D functions	51
cross_sections	51
paths2sections	51
path_scaling_sections	51
bezier_surface	51
bezier_smooth	51
midpt_smooth	51
in_polyline	51
Paths	52
arc_path	52
bspline_curve	52
bezier_curve	52
helix	52
golden_spiral	52
archimedean_spiral	52
sphere_spiral	52
torus_knot	52
Extrusion	52
box_extrude	52
ellipse_extrude	52
stereographic_extrude	53
rounded_extrude	53
bend_extrude	53
2D Shape	53
shape_taiwan	53
shape_arc	53

shape_pie	53
shape_circle	53
shape_ellipse	53
shape_square	53
shape_trapezium	53
shape_cyclicpolygon	53
shape_pentagram	53
shape_starburst	54
shape_superformula	54
shape_glue2circles	54
shape_path_extend	54
2D Shape extrusions	54
path_extrude	54
ring_extrude	54
helix_extrude	54
golden_spiral_extrude	54
archimedean_spiral_extrude	54
sphere_spiral_extrude	54
Utils	54
util/sub_str	55
New topic	55
New topic	55
New topic	55
New topic	55
New topic	55
New topic	55
New topic	55
New topic	55
New topic	55
New topic	55
BOSL Library	55
Nut Job Library	56

Introduction

OpenSCADHelp

OpenSCAD Help Project is a compilation of help information about OpenSCAD and some libraries.

Any programming language has two parts to learn. The language syntax and himself instructions and the common libraries of constructed instructions.

OpenSCAD Help is a compilation all this help in a unique site, with a unique form of search and found help about almost the most important things of OpenSCAD.

The libraries included in this project are the next:

MCAD

Components commonly used in designing and mocking up mechanical designs

<https://github.com/openscad/MCAD>

dotSCAD

Reduce the burden of 3D modeling in mathematics. <https://github.com/JustinSDK/dotSCAD>

BOSL

The Belfry OpenScad Library - A library of tools, shapes, and helpers to make OpenSCAD easier to use. <https://github.com/revarbat/BOSL>

Created with the Personal Edition of HelpNDoc: [Easily create Qt Help files](#)

What's new

2020 April 22th.

First version of PDF help file.
First version of HTML help site.
First version of EPUB help file.

Created with the Personal Edition of HelpNDoc: [Easy CHM and documentation editor](#)

Getting Started

OpenSCAD is a software for creating solid 3D CAD objects.
It is **free software** and available for **GNU/Linux**, Microsoft Windows and Mac OS X.

OpenSCAD focuses on the **CAD** aspects as opposition to artistic aspects. So it might be the application you are looking for when you are planning to create 3D models of machine parts.

OpenSCAD is not an interactive modeler. It is something like a 2D/3D-compiler that reads in a program file that describes the object and renders the model from this file. This gives you (the designer) full control over the modeling process. This enables you to easily change any step in the modeling process and make designs that are defined by configurable parameters.

OpenSCAD has two main operating modes:

- Preview is relatively fast using **3D graphics** and the **computer's GPU**, but is an approximation of the model and can produce **artifacts**; Preview uses **OpenCSG** and **OpenGL**.
- Render generates exact geometry and a fully **tessellated mesh**. It is not an approximation and as such it is often a lengthy process, taking minutes or hours for larger designs. Render uses **CGAL** as its geometry engine.

OpenSCAD provides two types of 3D modelling:

- **Constructive Solid Geometry (CSG)**
- extrusion of 2D primitives into 3D space.

OpenSCAD can be downloaded from <https://www.openscad.org/>. **OpenSCAD** is a software for creating solid 3D CAD objects.

It is **free software** and available for **GNU/Linux**, Microsoft Windows and Mac OS X.

Unlike most free software for creating 3D models (such as the well-known application **Blender**), OpenSCAD does not focus on the artistic aspects of 3D modelling, but instead focuses on the **CAD** aspects. So it might be the application you are looking for when you are planning to create 3D models of machine parts, but probably is not what you are looking for when you are more interested in creating computer-animated movies or organic life-like models.

OpenSCAD, unlike many CAD products, is not an interactive modeler. Instead it is something like a 2D/3D-compiler that reads in a program file that describes the object and renders the model from this file. This gives you (the designer) full control over the modelling process. This enables you to easily change any step in the modelling process and make designs that are defined by configurable parameters.

OpenSCAD has two main operating modes, *Preview* and *Render*. Preview is relatively fast using **3D graphics** and the **computer's GPU**, but is an approximation of the model and can produce **artifacts**; Preview uses **OpenCSG** and **OpenGL**. Render generates exact geometry and a fully **tessellated mesh**. It is not an approximation and as such it is often a lengthy process, taking minutes or hours for larger designs. Render uses **CGAL** as its geometry engine.

OpenSCAD provides two types of 3D modelling:

- **Constructive Solid Geometry (CSG)**
- extrusion of 2D primitives into 3D space.

Autocad DXF files are used as the data exchange format for 2D outlines. In addition to 2D paths for extrusion it is also possible to read design parameters from DXF files. Besides DXF files, OpenSCAD can read and create 3D models in the **STL** and **OFF** file formats.

OpenSCAD can be downloaded from <https://www.openscad.org/>. More information is available on the **mailing list**. OpenSCAD can also be tried online at <http://openscad.net/>, which is a partial port of OpenSCAD for the web. More information is available on the **mailing list**. OpenSCAD can also be tried online at <http://openscad.net/>, which is a partial port of OpenSCAD for the web.

Language Reference

The OpenSCAD Language

Introduction

Introduction

OpenSCAD is a 2D/3D and solid modeling program which is based on a Functional programming language used to create models that are previewed on the screen, and rendered into 3D mesh which allows the model to be exported in a variety of 2D/3D file formats.

A script in the OpenSCAD language is used to create 2D or 3D models. This script is a free format list of action statements.

```
object();
variable = value;
operator() action();
operator() {action(); action();}
operator() operator() {action(); action();}
operator() {operator() action();
              operator() {action(); action();}}
```

Objects

Objects are the building blocks for models, created by 2D and 3D primitives. Objects end in a semicolon ';'.

Actions

Action statements include creating objects using primitives and assigning values to variables. Action statements also end in a semicolon ';'.

Operators

Operators, or transformations, modify the location, color and other properties of objects. Operators use braces '{}' when their scope covers more than one action. More than one operator may be used for the same action or group of actions. Multiple operators are processed Right to Left, that is, the operator closest to the action is processed first. Operators do not end in semicolons ';', but the individual actions they contain do.

Examples

```
cube(5);
x = 4 + y;
rotate(40) square(5,10);
translate([10,5]) { circle(5); square(4); }
rotate(60) color("red") { circle(5); square(4); }
```

```
color("blue") { translate([5,3,0]) sphere(5); rotate([45,0,45]) { cylinder(10); cube([5,6,7]); } }
```

Created with the Personal Edition of HelpNDoc: [Easily create EPub books](#)

Comments

Comments

Comments are a way of leaving notes within the script, or code, (either to yourself or to future programmers) describing how the code works, or what it does. Comments are not evaluated by the compiler, and should not be used to describe self-evident code.

OpenSCAD uses C++-style comments:

```
// This is a comment

myvar = 10; // The rest of the line is a comment

/*
Multi-line comments
can span multiple lines.
*/
```

Created with the Personal Edition of HelpNDoc: [iPhone web sites made easy](#)

Values and Data Types

Values and Data Types

A value in OpenSCAD is either a Number (like 42), a Boolean (like true), a String (like "foo"), a Range (like [0: 1: 10]), a Vector (like [1,2,3]), or the Undefined value (undef).

Values can be stored in variables, passed as function arguments, and returned as function results.

[OpenSCAD is a dynamically typed language with a fixed set of data types. There are no type names, and no user defined types. Functions are not values. In fact, variables and functions occupy disjoint namespaces.]

Created with the Personal Edition of HelpNDoc: [Single source CHM, PDF, DOC and HTML Help creation](#)

Numbers

Numbers

Numbers are the most important type of value in OpenSCAD, and they are written in the familiar decimal notation used in other languages. Eg, -1, 42, 0.5, 2.99792458e+8.

[OpenSCAD does not support octal or hexadecimal notation for numbers.]

In addition to decimal numerals, the following names for special numbers are defined:

- **PI**

OpenSCAD has only a single kind of number, which is a 64 bit IEEE floating point number.

[OpenSCAD does not distinguish integers and floating point numbers as two different types, nor does it support complex numbers.]

Because OpenSCAD uses the IEEE floating point standard, there are a few deviations from the behavior of numbers in mathematics:

- We use binary floating point. A fractional number is not represented exactly unless the denominator is a power of 2. For example, 0.2 (2/10) does not have an exact internal representation, but 0.25 (1/4) and 0.125 (1/8) are represented exactly.
- The largest representable number is about 1e308. If a numeric result is too large, then the result can be infinity (printed as `inf` by `echo`).
- The smallest representable number is about -1e308. If a numeric result is too small, then the result can be -infinity (printed as `-inf` by `echo`).
- If a numeric result is invalid, then the result can be Not A Number (printed as **nan** by `echo`).
- If a non-zero numeric result is too close to zero to be representable, then the result is -0 if the result is negative, otherwise it is 0. Zero (0) and negative zero (-0) are treated as two distinct numbers by some of the math operations, and are printed differently by `'echo'`, although they compare as equal.

The constants **'inf'** and **'nan'** are not supported as numeric constants by OpenSCAD, even though you can compute numbers that are printed this way by `'echo'`. You can define variables with these values by using:

```
inf = 1e200 * 1e200;
nan = 0 / 0;
echo(inf, nan);
```

The value `'nan'` is the only OpenSCAD value that is not equal to any other value, including itself. Although you can test if a variable `'x'` has the undefined value using `'x == undef'`, you can't use `'x == 0/0'` to test if `x` is Not A Number. Instead, you must use `'x != x'` to test if `x` is nan.

Created with the Personal Edition of HelpNDoc: [Free PDF documentation generator](#)

Boolean Values

Boolean Values

Booleans are truth values. There are two Boolean values, namely `true` and `false`. A Boolean is passed as the argument to conditional statement `'if()'`, conditional operator `'?:'`, and logical operators `'!'` (not), `'&&'` (and), and `'||'` (or). In all of these contexts, you can actually pass any quantity. Most values are converted to `'true'` in a Boolean context, the values that count as `'false'` are:

- `false`
- `0` and `-0`
- `" "`
- `[]`

- undef

Note that `"false"` (the string), `[0]` (a numeric vector), `[[]]` (a vector containing an empty vector), `[false]` (a vector containing the Boolean value false) and `0/0` (Not A Number) all count as true.

Created with the Personal Edition of HelpNDoc: [Easily create EPub books](#)

Strings

Strings

A string is a sequence of zero or more unicode characters. String values are used to specify file names when importing a file, and to display text for debugging purposes when using `echo()`. Strings can also be used with the [text\(\) primitive](#).

A string literal is written as a sequence of characters enclosed in quotation marks `"`, like this: `""` (an empty string), or `"this is a string"`.

To include a `"` character in a string literal, use `\"`. To include a `\` character in a string literal, use `\\`. The following escape sequences beginning with `\` can be used within string literals:

- `\"` → `"`
- `\\` → `\`
- `\t` → tab
- `\n` → newline
- `\r` → carriage return
- `\u03a9` → `Ω` - see `text()` for further information on unicode characters

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

Ranges

Ranges

Ranges are used by [for\(\) loops](#) and [children\(\)](#). They have 2 varieties:

```
[<start>:<end>]
[<start>:<increment>:<end>]
```

Although enclosed in square brackets `[]`, they are not vectors. They use colons `:` for separators rather than commas.

```
r1 = [0:10];
r2 = [0.5:2.5:20];
echo(r1); // ECHO: [0: 1: 10]
echo(r2); // ECHO: [0.5: 2.5: 20]
```

You should avoid step values that cannot be represented exactly as binary floating point numbers. Integers are okay, as are fractional values whose denominator is a power of two. For example, 0.25 (1/4) and 0.125 (1/8) are safe, but 0.2 (2/10) should be avoided. The problem with these step values is that your range may have too many or too few elements, due to inexact arithmetic.

A missing `<increment>` defaults to 1.

A range in the form [*<start>*:*<end>*] with *<start>* greater than *<end>* generates a warning and is equivalent to [*<end>*: 1: *<start>*].

A range in the form [*<start>*:1:*<end>*] with *<start>* greater than *<end>* does not generate a warning and is equivalent to [].

The *<increment>* in a range may be negative.

Created with the Personal Edition of HelpNDoc: [Single source CHM, PDF, DOC and HTML Help creation](#)

The Undefined Value

The Undefined Value

The undefined value is a special value written as **undef**. It's the initial value of a variable that hasn't been assigned a value, and it is often returned as a result by functions or operations that are passed illegal arguments. Finally, **undef can be used as a null value**, equivalent to `null` or `NULL` in other programming languages.

All arithmetic expressions containing `undef` values evaluate as `undef`.

In logical expressions, `undef` is equivalent to `false`.

Relational operator expressions with `undef` evaluate as `false` except for `undef==undef` which is `true`.

Note that numeric operations may also return 'nan' (not-a-number) to indicate an illegal argument. For example, `0/false` is `undef`, but `0/0` is 'nan'. Relational operators like `<` and `>` return `false` if passed illegal arguments. Although `undef` is a language value, 'nan' is not.

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

Variables

Variables

OpenSCAD variables are created by a statement with a name or **identifier**, assignment via an expression and a semicolon. The role of arrays, found in many imperative languages, is handled in OpenSCAD via vectors.

```
var = 25;
xx = 1.25 * cos(50);
y = 2 * xx + var;
logic = true;
MyString = "This is a string";
a_vector = [1,2,3];
rr = a_vector[2]; // member of vector
rangel = [-1.5:0.5:3]; // for() loop range
xx = [0:5]; // alternate for() loop range
```

OpenSCAD is a **Functional** programming language, as such **variables** are bound to expressions and keep a single value during their entire lifetime due to the requirements of **referential transparency**. In **imperative languages**, such as C, the same behavior is seen as constants, which are typically contrasted with normal variables.

In other words OpenSCAD variables are more like constants, but with an important difference. If variables are assigned a value multiple times, only the last assigned value is used in all places in the code.

See further discussion at [Variables are set at compile-time, not run-time](#). This behavior is due to the need to supply variable input on the [command line](#), via the use of `-D variable=value` option. OpenSCAD currently places that assignment at the end of the source code, and thus must allow a variable's value to be changed for this purpose.

Values cannot be modified during run time; all variables are effectively constants that do not change. Each variable retains its last assigned value at compile time, in line with [Functional](#) programming languages. Unlike [Imperative](#) languages, such as C, OpenSCAD is not an iterative language, and as such **the concept of $x = x + 1$ is not valid**. Understanding this concept leads to understanding the beauty of OpenSCAD.

Variables can be assigned in any scope. Note that assignments are only valid within the scope in which they are defined - you are still not allowed to leak values to an outer scope. See [Scope of variables](#) for more details.

```
a=0;
if (a==0) {
    a=1; // but the value a=1 is confined to within the braces {}
}
```

Created with the Personal Edition of HelpNDoc: [Easy to use tool to create HTML Help files and Help web sites](#)

Undefined variable

Undefined variable

A non assigned variable has the special value **undef**. It could be tested in conditional expression, and returned by a function.

```
Example
echo("Variable a is ", a); // Variable a is undef
if (a == undef) {
    echo("Variable a is tested undefined"); // Variable a is tested
    undefined
}
```

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

Scope of variables

Scope of variables

When operators such as `translate()` and `color()` need to encompass more than one action (actions end in `;`), braces `{}` are needed to group the actions, creating a new, inner scope. When there is only one semicolon, braces are usually optional.

Each pair of braces creates a new scope inside the scope where they were used. New variables can be created within this new scope. New values can be given to variables which were created

in an outer scope. These variables and their values are also available to further inner scopes created within this scope, but are **not available** to any thing outside this scope. Variables still have only the last value assigned within a scope.

```

                                // scope 1
a = 6;                          // create a
echo(a, b);                     //           6, undef
translate([5, 0, 0]){          // scope 1.1
    a = 10;
    b = 16;                    // create b
    echo(a, b);                //           100, 16 a=10; was overridden by
later a = 100;
    color("blue") {            // scope 1.1.1
        echo(a, b);            //           100, 20
        cube();
        b=20;
    }                          // back to 1.1
    echo(a, b);                // 100, 16
    a = 100;                   // override a in 1.1
}                               // back to 1
echo(a, b);                     // 6, undef
color("red"){                   // scope 1.2
    cube();
    echo(a,b);                 // 6, undef
}                               // back to 1
echo(a, b);                     // 6, undef

```

//In this example, scopes 1 and 1.1 are outer scopes to 1.1.1 but 1.2 is not.

Anonymous scopes are not considered scopes:

```
{ angle = 45; } rotate(angle) square(10);
```

For() loops are not an exception to the rule about variables having only one value within a scope. A copy of loop contents is created for each pass. Each pass is given its own scope, allowing any variables to have unique values for that pass. No, you still can't do `a=a+1`;

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

Variables are set at compile-time, not at run-time

Variables are set at compile-time, not at run-time

Because OpenSCAD calculates its variable values at compile-time, not run-time, **the last variable assignment, within a scope apply everywhere in that scope, or inner scopes thereof**. It may be helpful to think of them as override-able constants rather than as variables.

```

// The value of 'a' reflects only the last set value
a = 0;
echo(a); // 5
a = 3;
echo(a); // 5 a = 5;

```

While this appears to be counter-intuitive, it allows you to do some interesting things: for instance, if you set up your shared library files to have default values defined as variables at their root level, when you include that file in your own code, you can 're-define' or override those

constants by simply assigning a new value to them.

Created with the Personal Edition of HelpNDoc: [News and information about help authoring tools and software](#)

Special Variables

Special Variables

Special variables provide an alternate means of passing arguments to modules and functions. **All variables starting with a '\$' are special variables**, similar to special variables in lisp. As such they are more dynamic than regular variables. (for more details see [Other Language Features](#))

Created with the Personal Edition of HelpNDoc: [Benefits of a Help Authoring Tool](#)

Vectors

Vectors

A vector is a sequence of zero or more OpenSCAD values. Vectors are a collection (or list or table) of numeric or boolean values, variables, vectors, strings or any combination thereof. They can also be expressions which evaluate to one of these.

Vectors handle the role of arrays found in many imperative languages. The information here also applies to lists and tables which use vectors for their data.

A vector has square brackets, `[]` enclosing zero or more items (elements or members), separated by commas. A vector can contain vectors, which contain vectors, etc.

examples

```
[1, 2, 3]
[a, 5, b]
[]
[5.643]
["a", "b", "string"]
[[1, r], [x, y, z, 4, 5]]
[3, 5, [6,7], [[8, 9], [10, [11, 12], 13], c, "string"]
[4/3, 6*1.5, cos(60)]
```

use in OpenSCAD:

```
cube( [width, depth, height]); // optional spaces shown for clarity
translate( [x, y, z] )
polygon( [ [x0, y0], [x1, y1], [x2, y2] ] );
```

creation

Vectors are created by writing the list of elements, separated by commas, and enclosed in square brackets. Variables are replaced by their values.

```

cube([10, 15, 20]);
a1 = [1, 2, 3];
a2 = [4, 5];
a3 = [6, 7, 8, 9];
b = [a1, a2, a3]; // [ [1,2,3], [4,5], [6,7,8,9] ] note increased nesting depth

```

elements within vectors

Elements within vectors are numbered from 0 to n-1 where n is the length returned by `len()`. Address elements within vectors with the following notation:

```

e[5]           // element no 5 (sixth) at 1st nesting level
e[5][2]        // element 2 of element 5 2nd nesting level
e[5][2][0]     // element 0 of 2 of 5 3rd nesting level
e[5][2][0][1] // element 1 of 0 of 2 of 5 4th nesting level

```

example elements with lengths from `len()`

```
e = [ [1], [], [3, 4, 5], "string", "x", [[10, 11], [12, 13, 14], [[15, 16], [17]]] ]; // length 6
```

address	length	element
e[0]	1	[1]
e[1]	0	[]
e[5]	3	[[10,11], [12,13,14], [[15,16],[17]]]
e[5][1]	3	[12, 13, 14]
e[5][2]	2	[[15, 16], [17]]
e[5][2][0]	2	[15, 16]
e[5][2][0][1]	undef	16
e[3]	6	"string"
e[3][2]	1	"r"
s = [2,0,5];	a = 2;	
s[a]	undef	5
e[s[a]]	3	[[10, 11], [12, 13, 14], [[15, 16], [17]]]

alternate dot notation

The first three elements of a vector can be accessed with an alternate dot notation:

```

e.x //equivalent to e[0]
e.y //equivalent to e[1]
e.z //equivalent to e[2]

```

Created with the Personal Edition of HelpNDoc: [Produce Kindle eBooks easily](#)

Vector Operators (concat & len)

Vector Operators (concat & len)

concat

`concat()` combines the elements of 2 or more vectors into a single vector. No change in nesting

level is made.

```
vector1 = [1, 2, 3]; vector2 = [4]; vector3 = [5,6];
new_vector = concat(vector1, vector2, vector3); // [1, 2, 3, 4, 5, 6]

string_vector = concat("abc", "def"); // ["abc", "def"]
one_string = str(string_vector[0], string_vector[1]); // "abcdef"
```

len

len() returns the length of vectors or strings. Indices of elements are from [0] to [length-1].

vector

- Returns the number of elements at this level.
- Single values, which are **not** vectors, return **undef**.

string

- Returns the number of characters in string.

```
a = [1, 2, 3]; echo(len(a)); // 3
```

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

Matrix

Matrix

A matrix is a vector of vectors.

Example which defines a 2D rotation matrix

```
mr = [
    [cos(angle), -sin(angle)],
    [sin(angle), cos(angle)]
];
```

Created with the Personal Edition of HelpNDoc: [Free CHM Help documentation generator](#)

Getting input

Getting input

Now we have variables, it would be nice to be able to get input into them instead of setting the values from code. There are a few functions to read data from DXF files, or you can set a variable with the -D switch on the command line.

Getting a point from a drawing

Getting a point is useful for reading an origin point in a 2D view in a technical drawing. The function dxf_cross reads the intersection of two lines on a layer you specify and returns the intersection point. This means that the point must be given with two lines in the DXF file, and not a point entity.

```
OriginPoint = dxf_cross(file = "drawing.dxf",
                        layer = "SCAD.Origin",
```

```
origin = [0, 0], scale = 1);
```

Getting a dimension value

You can read dimensions from a technical drawing. This can be useful to read a rotation angle, an extrusion height, or spacing between parts. In the drawing, create a dimension that does not show the dimension value, but an identifier. To read the value, you specify this identifier from your program:

```
TotalWidth = dxf_dim(file    = "drawing.dxf",
                      name    = "TotalWidth",
                      layer   = "SCAD.Origin",
                      origin   = [0, 0],
                      scale    = 1);
```

For a nice example of both functions, see Example009 and the image on the [homepage of OpenSCAD](#).

Created with the Personal Edition of HelpNDoc: [Easily create EBooks](#)

3D Objects

Created with the Personal Edition of HelpNDoc: [Generate Kindle eBooks with ease](#)

Primitive Solids

Created with the Personal Edition of HelpNDoc: [Produce Kindle eBooks easily](#)

cube

Created with the Personal Edition of HelpNDoc: [Free Web Help generator](#)

sphere

Created with the Personal Edition of HelpNDoc: [Easily create EPub books](#)

cylinder

Created with the Personal Edition of HelpNDoc: [Full-featured Documentation generator](#)

polyhedron

Created with the Personal Edition of HelpNDoc: [Easy EBook and documentation generator](#)

Debugging polyhedra

Created with the Personal Edition of HelpNDoc: [Easily create Web Help sites](#)

Mis-ordered faces

Created with the Personal Edition of HelpNDoc: [Easily create Help documents](#)

Point repetitions in a polyhedron point list

Created with the Personal Edition of HelpNDoc: [News and information about help authoring tools and software](#)

3D to 2D Projection

Created with the Personal Edition of HelpNDoc: [Full-featured Help generator](#)

2D Objects

Created with the Personal Edition of HelpNDoc: [Single source CHM, PDF, DOC and HTML Help creation](#)

square

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

circle

Created with the Personal Edition of HelpNDoc: [Easily create Web Help sites](#)

ellipse

Created with the Personal Edition of HelpNDoc: [Write EPub books for the iPad](#)

regular polygon

Created with the Personal Edition of HelpNDoc: [Full-featured Help generator](#)

polygon

Created with the Personal Edition of HelpNDoc: [Free HTML Help documentation generator](#)

import_dxf

Created with the Personal Edition of HelpNDoc: [Free PDF documentation generator](#)

Text

Created with the Personal Edition of HelpNDoc: [Full-featured EPub generator](#)

Using Fonts & Styles

Created with the Personal Edition of HelpNDoc: [Free EPub producer](#)

Alignment

Created with the Personal Edition of HelpNDoc: [Full-featured Documentation generator](#)

Vertical Alignment

Created with the Personal Edition of HelpNDoc: [What is a Help Authoring tool?](#)

Horizontal Alignment

Created with the Personal Edition of HelpNDoc: [Full-featured multi-format Help generator](#)

3D Text

Created with the Personal Edition of HelpNDoc: [Easy CHM and documentation editor](#)

3D to 2D Projection

Created with the Personal Edition of HelpNDoc: [Free help authoring tool](#)

2D to 3D Projection

Created with the Personal Edition of HelpNDoc: [Generate EPub eBooks with ease](#)

Linear Extrude

Created with the Personal Edition of HelpNDoc: [Full-featured EPub generator](#)

Usage

Created with the Personal Edition of HelpNDoc: [Free EPub producer](#)

Twist

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

Center

Created with the Personal Edition of HelpNDoc: [Free Web Help generator](#)

Mesh Refinement

Created with the Personal Edition of HelpNDoc: [Full-featured EPub generator](#)

Scale

Created with the Personal Edition of HelpNDoc: [Free iPhone documentation generator](#)

Rotate Extrude

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

Parameters

Created with the Personal Edition of HelpNDoc: [Full-featured Help generator](#)

Usage

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

Examples

Created with the Personal Edition of HelpNDoc: [Free EBook and documentation generator](#)

Mesh Refinement

Created with the Personal Edition of HelpNDoc: [Easily create PDF Help documents](#)

Extruding a Polygon

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

Description of extrude Parameters

Created with the Personal Edition of HelpNDoc: [iPhone web sites made easy](#)

Extrude parameters for all extrusion modes

Created with the Personal Edition of HelpNDoc: [Easily create Qt Help files](#)

Extrude parameters for linear extrusion only

Created with the Personal Edition of HelpNDoc: [News and information about help authoring tools and software](#)

Transforms

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

Basic concept

Created with the Personal Edition of HelpNDoc: [Create iPhone web-based documentation](#)

Advanced concept

Created with the Personal Edition of HelpNDoc: [Free CHM Help documentation generator](#)

scale

Created with the Personal Edition of HelpNDoc: [Produce electronic books easily](#)

resize

Created with the Personal Edition of HelpNDoc: [What is a Help Authoring tool?](#)

rotate

Created with the Personal Edition of HelpNDoc: [Produce electronic books easily](#)

Rotation rule Help

Created with the Personal Edition of HelpNDoc: [Easily create Help documents](#)

translate

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

mirror

Created with the Personal Edition of HelpNDoc: [Full-featured Help generator](#)

Function signature:

Created with the Personal Edition of HelpNDoc: [Easily create HTML Help documents](#)

Examples

Created with the Personal Edition of HelpNDoc: [Free Qt Help documentation generator](#)

multimatrix

Created with the Personal Edition of HelpNDoc: [Easily create EBooks](#)

More?

Created with the Personal Edition of HelpNDoc: [Create HTML Help, DOC, PDF and print manuals from 1 single source](#)

color

Created with the Personal Edition of HelpNDoc: [Full-featured EBook editor](#)

function signature

Created with the Personal Edition of HelpNDoc: [Free help authoring tool](#)

Example

Created with the Personal Edition of HelpNDoc: [Free PDF documentation generator](#)

Example 2

Created with the Personal Edition of HelpNDoc: [Free Kindle producer](#)

offset

Created with the Personal Edition of HelpNDoc: [Benefits of a Help Authoring Tool](#)

minkowski

Created with the Personal Edition of HelpNDoc: [Write eBooks for the Kindle](#)

hull

Created with the Personal Edition of HelpNDoc: [Easily create Help documents](#)

Combining transformations

Created with the Personal Edition of HelpNDoc: [Write eBooks for the Kindle](#)

Boolean Combinations

Created with the Personal Edition of HelpNDoc: [Write EPub books for the iPad](#)

Boolean overview

Created with the Personal Edition of HelpNDoc: [Single source CHM, PDF, DOC and HTML Help creation](#)

2D examples

Created with the Personal Edition of HelpNDoc: [Easily create HTML Help documents](#)

3D examples

Created with the Personal Edition of HelpNDoc: [News and information about help authoring tools and software](#)

union

Created with the Personal Edition of HelpNDoc: [Easy EPub and documentation editor](#)

difference

Created with the Personal Edition of HelpNDoc: [Free EPub and documentation generator](#)

difference with multiple children

Created with the Personal Edition of HelpNDoc: [Free Web Help generator](#)

intersection

Created with the Personal Edition of HelpNDoc: [iPhone web sites made easy](#)

render

Created with the Personal Edition of HelpNDoc: [Write EPub books for the iPad](#)

Other Functions and Operators

Created with the Personal Edition of HelpNDoc: [Full-featured Documentation generator](#)

Condition and Iterator functions

Created with the Personal Edition of HelpNDoc: [Free Kindle producer](#)

For loop

Created with the Personal Edition of HelpNDoc: [Easy to use tool to create HTML Help files and Help web sites](#)

Intersection For loop

Created with the Personal Edition of HelpNDoc: [Free EPub and documentation generator](#)

if statement

Created with the Personal Edition of HelpNDoc: [Full-featured Kindle eBooks generator](#)

else if

Created with the Personal Edition of HelpNDoc: [Full-featured Help generator](#)

Conditional ?

Created with the Personal Edition of HelpNDoc: [Easy EPub and documentation editor](#)

Recursive function calls

Created with the Personal Edition of HelpNDoc: [Easily create iPhone documentation](#)

Assign Statement

Created with the Personal Edition of HelpNDoc: [Free EPub producer](#)

Let Statement

Created with the Personal Edition of HelpNDoc: [Free EPub producer](#)

Mathematical Operators

Created with the Personal Edition of HelpNDoc: [Full-featured multi-format Help generator](#)

Scalar Arithmetical Operators

Created with the Personal Edition of HelpNDoc: [Easily create CHM Help documents](#)

Relational Operators

Created with the Personal Edition of HelpNDoc: [Create HTML Help, DOC, PDF and print manuals from 1 single source](#)

Logical Operators

Created with the Personal Edition of HelpNDoc: [Produce electronic books easily](#)

Conditional Operator

Created with the Personal Edition of HelpNDoc: [Single source CHM, PDF, DOC and HTML Help creation](#)

Vector-Number Operator

Created with the Personal Edition of HelpNDoc: [Qt Help documentation made easy](#)

Vector Operators

Created with the Personal Edition of HelpNDoc: [iPhone web sites made easy](#)

Vector Dot-Product Operator

Created with the Personal Edition of HelpNDoc: [Full-featured multi-format Help generator](#)

Matrix Multiplication

Created with the Personal Edition of HelpNDoc: [Produce electronic books easily](#)

Mathematical Functions

Created with the Personal Edition of HelpNDoc: [Free Qt Help documentation generator](#)

Trigonometric Functions

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

cos

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

sin

Created with the Personal Edition of HelpNDoc: [Full-featured Documentation generator](#)

tan

Created with the Personal Edition of HelpNDoc: [Easily create EPub books](#)

acos

Created with the Personal Edition of HelpNDoc: [Free EBook and documentation generator](#)

asin

Created with the Personal Edition of HelpNDoc: [iPhone web sites made easy](#)

atan

Created with the Personal Edition of HelpNDoc: [Free Qt Help documentation generator](#)

atan2

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

Other Mathematical Functions

Created with the Personal Edition of HelpNDoc: [Generate Kindle eBooks with ease](#)

abs

Created with the Personal Edition of HelpNDoc: [Full-featured multi-format Help generator](#)

ceil

Created with the Personal Edition of HelpNDoc: [Free CHM Help documentation generator](#)

concat

Created with the Personal Edition of HelpNDoc: [Free help authoring environment](#)

cross

Created with the Personal Edition of HelpNDoc: [Free iPhone documentation generator](#)

exp

Created with the Personal Edition of HelpNDoc: [Free Web Help generator](#)

floor

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

in

Created with the Personal Edition of HelpNDoc: [Benefits of a Help Authoring Tool](#)

len

Created with the Personal Edition of HelpNDoc: [Easily create CHM Help documents](#)

let

Created with the Personal Edition of HelpNDoc: [What is a Help Authoring tool?](#)

log

Created with the Personal Edition of HelpNDoc: [Easy EPub and documentation editor](#)

lookup

Created with the Personal Edition of HelpNDoc: [Full-featured multi-format Help generator](#)

max

Created with the Personal Edition of HelpNDoc: [Benefits of a Help Authoring Tool](#)

min

Created with the Personal Edition of HelpNDoc: [Full-featured multi-format Help generator](#)

norm

Created with the Personal Edition of HelpNDoc: [Write EPub books for the iPad](#)

pow

Created with the Personal Edition of HelpNDoc: [Free EPub producer](#)

rands

Created with the Personal Edition of HelpNDoc: [Free help authoring environment](#)

round

Created with the Personal Edition of HelpNDoc: [What is a Help Authoring tool?](#)

sign

Created with the Personal Edition of HelpNDoc: [What is a Help Authoring tool?](#)

sqrt

Created with the Personal Edition of HelpNDoc: [Easy to use tool to create HTML Help files and Help web sites](#)

Infinities and NaNs

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

String Functions

Created with the Personal Edition of HelpNDoc: [What is a Help Authoring tool?](#)

str

Created with the Personal Edition of HelpNDoc: [Write EPub books for the iPad](#)

chr

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

ord

Created with the Personal Edition of HelpNDoc: [Free PDF documentation generator](#)

Also see [search\(\)](#)

Created with the Personal Edition of HelpNDoc: [Write eBooks for the Kindle](#)

List Comprehensions

Created with the Personal Edition of HelpNDoc: [Single source CHM, PDF, DOC and HTML Help creation](#)

Basic Syntax

Created with the Personal Edition of HelpNDoc: [Free HTML Help documentation generator](#)

Multiple generator expressions

Created with the Personal Edition of HelpNDoc: [Free Web Help generator](#)

for

Created with the Personal Edition of HelpNDoc: [Easily create EPub books](#)

each

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

if

Created with the Personal Edition of HelpNDoc: [Full-featured EBook editor](#)

if/else

Created with the Personal Edition of HelpNDoc: [Easily create EPub books](#)

let

Created with the Personal Edition of HelpNDoc: [Write EPub books for the iPad](#)

Nested loops

Created with the Personal Edition of HelpNDoc: [Create iPhone web-based documentation](#)

Advanced Examples

Created with the Personal Edition of HelpNDoc: [Easily create EPub books](#)

Generating vertices for a polygon

Created with the Personal Edition of HelpNDoc: [Free EPub producer](#)

Flattening a nested vector

Created with the Personal Edition of HelpNDoc: [Write eBooks for the Kindle](#)

Sorting a vector

Created with the Personal Edition of HelpNDoc: [Free Web Help generator](#)

Selecting elements of a vector

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

Concatenating two vectors

Created with the Personal Edition of HelpNDoc: [What is a Help Authoring tool?](#)

Other Language Features

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

Special Variables

Created with the Personal Edition of HelpNDoc: [Full-featured EPub generator](#)

\$fa, \$fs and \$fn

Created with the Personal Edition of HelpNDoc: [Easily create HTML Help documents](#)

\$t

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

\$vpr, \$vpt and \$vpd

Created with the Personal Edition of HelpNDoc: [Create HTML Help, DOC, PDF and print manuals from 1 single source](#)

\$preview

Created with the Personal Edition of HelpNDoc: [Full-featured Documentation generator](#)

Echo Statements

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

Usage examples

Created with the Personal Edition of HelpNDoc: [Easily create PDF Help documents](#)

Rounding examples

Created with the Personal Edition of HelpNDoc: [Full-featured EBook editor](#)

Small and large Numbers

Created with the Personal Edition of HelpNDoc: [Easily create PDF Help documents](#)

HTML

Created with the Personal Edition of HelpNDoc: [Free Qt Help documentation generator](#)

Echo Function

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

Render

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

Surface

Created with the Personal Edition of HelpNDoc: [Free HTML Help documentation generator](#)

Text file format

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

Images

Created with the Personal Edition of HelpNDoc: [Easily create HTML Help documents](#)

Examples

Created with the Personal Edition of HelpNDoc: [Easily create EBooks](#)

Search

Created with the Personal Edition of HelpNDoc: [Easy CHM and documentation editor](#)

Search Usage

Created with the Personal Edition of HelpNDoc: [Full-featured Help generator](#)

Search Arguments

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

Search Usage Examples

Created with the Personal Edition of HelpNDoc: [Full-featured EBook editor](#)

Index values return as list

Created with the Personal Edition of HelpNDoc: [Easily create CHM Help documents](#)

Search on different column; return index values

Created with the Personal Edition of HelpNDoc: [Produce Kindle eBooks easily](#)

Search on list of values

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

Search on list of strings

Created with the Personal Edition of HelpNDoc: [Free EPub producer](#)

Getting the right result

Created with the Personal Edition of HelpNDoc: [Free EPub and documentation generator](#)

OpenSCAD Version

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

parent_module(n) and \$parent_modules

Created with the Personal Edition of HelpNDoc: [Easily create Qt Help files](#)

assert

Created with the Personal Edition of HelpNDoc: [Easily create iPhone documentation](#)

Example

Created with the Personal Edition of HelpNDoc: [Single source CHM, PDF, DOC and HTML Help creation](#)

Checking parameters

Created with the Personal Edition of HelpNDoc: [Produce Kindle eBooks easily](#)

Adding message

Created with the Personal Edition of HelpNDoc: [News and information about help authoring tools and software](#)

Using assertions in functions

Created with the Personal Edition of HelpNDoc: [Free PDF documentation generator](#)

User Defined Functions and Modules

Created with the Personal Edition of HelpNDoc: [Produce Kindle eBooks easily](#)

Introduction

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

Scope

Created with the Personal Edition of HelpNDoc: [Free iPhone documentation generator](#)

Functions

Created with the Personal Edition of HelpNDoc: [Full-featured Kindle eBooks generator](#)

Recursive Functions

Created with the Personal Edition of HelpNDoc: [Qt Help documentation made easy](#)

Function Literals

Created with the Personal Edition of HelpNDoc: [Generate EPub eBooks with ease](#)

Modules

Created with the Personal Edition of HelpNDoc: [Easy to use tool to create HTML Help files and Help web sites](#)

Object modules

Created with the Personal Edition of HelpNDoc: [Benefits of a Help Authoring Tool](#)

Operator Modules

Created with the Personal Edition of HelpNDoc: [Easily create PDF Help documents](#)

Children

Created with the Personal Edition of HelpNDoc: [Free help authoring environment](#)

Further Module Examples

Created with the Personal Edition of HelpNDoc: [Easily create PDF Help documents](#)

Recursive Modules

Created with the Personal Edition of HelpNDoc: [Create iPhone web-based documentation](#)

Overwriting built-in modules

Created with the Personal Edition of HelpNDoc: [Easily create Help documents](#)

Overwriting built functions

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

Debugging Aids

Created with the Personal Edition of HelpNDoc: [Free Qt Help documentation generator](#)

Advanced Concept

Created with the Personal Edition of HelpNDoc: [Free HTML Help documentation generator](#)

Background Modifier

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

Debug Modifier

Created with the Personal Edition of HelpNDoc: [Benefits of a Help Authoring Tool](#)

Root Modifier

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

Disable Modifier

Created with the Personal Edition of HelpNDoc: [Free Qt Help documentation generator](#)

Echo Statements

Created with the Personal Edition of HelpNDoc: [Free Web Help generator](#)

External Libraries and code files

Created with the Personal Edition of HelpNDoc: [Easily create Qt Help files](#)

Use and Include

Created with the Personal Edition of HelpNDoc: [Free help authoring environment](#)

Directory separators

Created with the Personal Edition of HelpNDoc: [Easy to use tool to create HTML Help files and Help web sites](#)

Variables

Created with the Personal Edition of HelpNDoc: [Free Kindle producer](#)

Scope of Variables

Created with the Personal Edition of HelpNDoc: [Full-featured Kindle eBooks generator](#)

Overwriting variables

Created with the Personal Edition of HelpNDoc: [What is a Help Authoring tool?](#)

Example "Ring-Library"

Created with the Personal Edition of HelpNDoc: [Easily create Help documents](#)

Nested Include and Use

Created with the Personal Edition of HelpNDoc: [Create HTML Help, DOC, PDF and print manuals from 1 single source](#)

Import

Created with the Personal Edition of HelpNDoc: [Create iPhone web-based documentation](#)

Parameters

Created with the Personal Edition of HelpNDoc: [Single source CHM, PDF, DOC and HTML Help creation](#)

Convexity

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

Notes

Created with the Personal Edition of HelpNDoc: [Free EPub and documentation generator](#)

Import DXF

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

Import STL

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

surface

Created with the Personal Edition of HelpNDoc: [Produce Kindle eBooks easily](#)

Parameters

Created with the Personal Edition of HelpNDoc: [Easily create eBooks](#)

Text file Format

Created with the Personal Edition of HelpNDoc: [Free help authoring tool](#)

Images

Created with the Personal Edition of HelpNDoc: [Full-featured multi-format Help generator](#)

Examples

Created with the Personal Edition of HelpNDoc: [Easily create iPhone documentation](#)

MCAD Library

MCAD Library

This library contains components commonly used in designing and mocking up mechanical designs. It is currently unfinished and you can expect some API changes, however many things are already working.

This library was created by various authors as named in the individual files' comments. All the files are

licensed under the LGPL 2.1 (see <http://creativecommons.org/licenses/LGPL/2.1/> or the included file lgpl-2.1.txt), some of them allow distribution under more permissive terms (as described in the files' comments).

Usage

You can import these files in your scripts with use <MCAD/filename.scad>, where 'filename' is one of the files listed below like 'motors' or 'servos'. Some files include useful constants which will be available with include <MCAD/filename.scad>, which should be safe to use on all included files (ie. no top level code should create geometry).

(There is a bug/feature that prevents including constants from files that "include" other files - see the openscad mailing list archives for more details. Since the maintainers aren't very responsive, may have to work around this somehow)

If you host your project in git, you can do git submodule add URL PATH in your repo to import this library as a git submodule for easy usage. Then you need to do a git submodule update --init after cloning. When you want to update the submodule, do cd PATH; git checkout master; git pull. See git help submodule for more info.

Currently Provided Tools:

```
regular_shapes.scad
    regular polygons, ie. 2D
    regular polyhedrons, ie. 3D
involute_gears.scad (http://www.thingiverse.com/thing:3575):
    gear()
    bevel_gear()
    bevel_gear_pair()
gears.scad (Old version):
    gear(number_of_teeth, circular_pitch OR diametrial_pitch, pressure_angle OPTIONAL,
clearance OPTIONAL)
motors.scad:
    stepper_motor_mount(nema_standard, slide_distance OPTIONAL, mochup OPTIONAL)
```

Tools (alpha and beta quality):

```
nuts_and_bolts.scad: for creating metric and imperial bolt/nut holes
bearing.scad: standard/custom bearings
screw.scad: screws and augers
materials.scad: color definitions for different materials
stepper.scad: NEMA standard stepper outlines
servos.scad: servo outlines
boxes.scad: box with rounded corners
triangles.scad: simple triangles
3d_triangle.scad: more advanced triangles
```

Very generally useful functions and constants:

```
math.scad: general math functions
constants.scad: mathematical constants
curves.scad: mathematical functions defining curves
units.scad: easy metric units
utilities.scad: geometric funtions and misc. useful stuff
teardrop.scad (http://www.thingiverse.com/thing:3457): parametric teardrop module
shapes.scad: DEPRECATED simple shapes by Catarina Mota
polyholes.scad: holes that should come out well when printed
```

Other:

alphabet_block.scad
 bitmap.scad
 letter_necklace.scad
 name_tag.scad
 height_map.scad
 trochoids.scad
 libtriangles.scad
 layouts.scad
 transformations.scad
 2Dshapes.scad
 gridbeam.scad
 fonts.scad
 unregular_shapes.scad
 metric_fasteners.scad
 lego_compatibility.scad
 multiply.scad
 hardware.scad

External utils that generate and process openscad code:

openscad_testing.py: testing code, see below
 openscad_utils.py: code for scraping function names etc.

Created with the Personal Edition of HelpNDoc: [Free Kindle producer](#)

regular_shapes.scad

Created with the Personal Edition of HelpNDoc: [Produce electronic books easily](#)

2D regular shapes

Created with the Personal Edition of HelpNDoc: [Create iPhone web-based documentation](#)

regular_polygon(sides, radius)

Created with the Personal Edition of HelpNDoc: [Produce Kindle eBooks easily](#)

n-gons 2D shapes

Created with the Personal Edition of HelpNDoc: [Easily create EPub books](#)

triangle(radius)

Created with the Personal Edition of HelpNDoc: [Free CHM Help documentation generator](#)

pentagon(radius)

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

hexagon(radius)

Created with the Personal Edition of HelpNDoc: [Easy EBook and documentation generator](#)

heptagon(radius)

Created with the Personal Edition of HelpNDoc: [Easily create CHM Help documents](#)

octagon(radius)

Created with the Personal Edition of HelpNDoc: [Create HTML Help, DOC, PDF and print manuals from 1 single source](#)

nonagon(radius)

Created with the Personal Edition of HelpNDoc: [Free HTML Help documentation generator](#)

decagon(radius)

Created with the Personal Edition of HelpNDoc: [News and information about help authoring tools and software](#)

hendecagon(radius)

Created with the Personal Edition of HelpNDoc: [Free HTML Help documentation generator](#)

dodecagon(radius)

Created with the Personal Edition of HelpNDoc: [Easy EBook and documentation generator](#)

ring(inside_diameter, thickness)

Created with the Personal Edition of HelpNDoc: [Free HTML Help documentation generator](#)

ellipse(width, height)

Created with the Personal Edition of HelpNDoc: [Full-featured Help generator](#)

egg_outline(width, thickness)

Created with the Personal Edition of HelpNDoc: [Easily create Qt Help files](#)

3D regular shapes

Created with the Personal Edition of HelpNDoc: [Easy CHM and documentation editor](#)

cone

Created with the Personal Edition of HelpNDoc: [Produce electronic books easily](#)

oval_prism

Created with the Personal Edition of HelpNDoc: [Produce Kindle eBooks easily](#)

oval_tube

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

cylinder_tube

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

triangle_prism

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

triangle_tube

Created with the Personal Edition of HelpNDoc: [Create iPhone web-based documentation](#)

pentagon_prism

Created with the Personal Edition of HelpNDoc: [Easily create iPhone documentation](#)

pentagon_tube

Created with the Personal Edition of HelpNDoc: [Full-featured Documentation generator](#)

hexagon_prism

Created with the Personal Edition of HelpNDoc: [Free help authoring tool](#)

hexagon_tube

Created with the Personal Edition of HelpNDoc: [Easily create EPub books](#)

heptagon_prism

Created with the Personal Edition of HelpNDoc: [Easily create CHM Help documents](#)

heptagon_tube

Created with the Personal Edition of HelpNDoc: [Free CHM Help documentation generator](#)

octagon_prism

Created with the Personal Edition of HelpNDoc: [Benefits of a Help Authoring Tool](#)

octagon_tube

Created with the Personal Edition of HelpNDoc: [Create HTML Help, DOC, PDF and print manuals from 1 single source](#)

nonagon_prism

Created with the Personal Edition of HelpNDoc: [Full-featured multi-format Help generator](#)

`decagon_prism`

Created with the Personal Edition of HelpNDoc: [Free iPhone documentation generator](#)

`hendecagon_prism`

Created with the Personal Edition of HelpNDoc: [Easy to use tool to create HTML Help files and Help web sites](#)

`dodecagon_prism`

Created with the Personal Edition of HelpNDoc: [Free HTML Help documentation generator](#)

`torus`

Created with the Personal Edition of HelpNDoc: [Produce electronic books easily](#)

`torus2`

Created with the Personal Edition of HelpNDoc: [Easy EBook and documentation generator](#)

`oval_torus`

Created with the Personal Edition of HelpNDoc: [Write eBooks for the Kindle](#)

`triangle_pyramid`

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

`square_pyramid`

Created with the Personal Edition of HelpNDoc: [iPhone web sites made easy](#)

`egg`

Created with the Personal Edition of HelpNDoc: [Free iPhone documentation generator](#)

`involute_gears.scad`

Created with the Personal Edition of HelpNDoc: [Free iPhone documentation generator](#)

`bevel_gear_pair()`

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

`bevel_gear()`

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

gear()

Created with the Personal Edition of HelpNDoc: [Free PDF documentation generator](#)

Tests

Created with the Personal Edition of HelpNDoc: [News and information about help authoring tools and software](#)

test_gears()

Created with the Personal Edition of HelpNDoc: [Easily create iPhone documentation](#)

test_meshing_double_helix()

Created with the Personal Edition of HelpNDoc: [Full-featured EBook editor](#)

test_bevel_gear()

Created with the Personal Edition of HelpNDoc: [Single source CHM, PDF, DOC and HTML Help creation](#)

test_bevel_gear_pair()

Created with the Personal Edition of HelpNDoc: [Free Web Help generator](#)

test_backlash()

Created with the Personal Edition of HelpNDoc: [News and information about help authoring tools and software](#)

dotSCAD Library

Created with the Personal Edition of HelpNDoc: [Qt Help documentation made easy](#)

2D modules

Created with the Personal Edition of HelpNDoc: [Free EBook and documentation generator](#)

arc

Created with the Personal Edition of HelpNDoc: [Easily create EBooks](#)

pie

Created with the Personal Edition of HelpNDoc: [Free PDF documentation generator](#)

rounded_square

Created with the Personal Edition of HelpNDoc: [Easy EBook and documentation generator](#)

line2d

Created with the Personal Edition of HelpNDoc: [Free EPub producer](#)

polyline2d

Created with the Personal Edition of HelpNDoc: [Easy EPub and documentation editor](#)

hull_polyline2d

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

hexagons

Created with the Personal Edition of HelpNDoc: [Easily create EBooks](#)

polytransversals

Created with the Personal Edition of HelpNDoc: [Produce electronic books easily](#)

multi_line_text

Created with the Personal Edition of HelpNDoc: [Easy EBook and documentation generator](#)

voronoi2d

Created with the Personal Edition of HelpNDoc: [Qt Help documentation made easy](#)

3D modules

Created with the Personal Edition of HelpNDoc: [Create iPhone web-based documentation](#)

rounded_cube

Created with the Personal Edition of HelpNDoc: [Easy EPub and documentation editor](#)

rounded_cylinder

Created with the Personal Edition of HelpNDoc: [Easy EPub and documentation editor](#)

crystal_ball

Created with the Personal Edition of HelpNDoc: [Easily create EPub books](#)

line3d

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

polyline3d

Created with the Personal Edition of HelpNDoc: [Easy EBook and documentation generator](#)

hull_polyline3d

Created with the Personal Edition of HelpNDoc: [iPhone web sites made easy](#)

function_grapher

Created with the Personal Edition of HelpNDoc: [Free help authoring environment](#)

sweep

Created with the Personal Edition of HelpNDoc: [Free EBook and documentation generator](#)

loft

Created with the Personal Edition of HelpNDoc: [Full-featured Kindle eBooks generator](#)

starburst

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

voronoi3d

Created with the Personal Edition of HelpNDoc: [Produce Kindle eBooks easily](#)

Transformations

Created with the Personal Edition of HelpNDoc: [Easy EPub and documentation editor](#)

along_width

Created with the Personal Edition of HelpNDoc: [Free Qt Help documentation generator](#)

hollow_out

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

bend

Created with the Personal Edition of HelpNDoc: [Free Kindle producer](#)

shear

Created with the Personal Edition of HelpNDoc: [Produce Kindle eBooks easily](#)

2D functions

Created with the Personal Edition of HelpNDoc: [Easily create PDF Help documents](#)

in_shape

Created with the Personal Edition of HelpNDoc: [Create iPhone web-based documentation](#)

bijection_offset

Created with the Personal Edition of HelpNDoc: [Full-featured Documentation generator](#)

trim_shape

Created with the Personal Edition of HelpNDoc: [Create iPhone web-based documentation](#)

triangulate

Created with the Personal Edition of HelpNDoc: [Generate EPub eBooks with ease](#)

contours

Created with the Personal Edition of HelpNDoc: [Generate Kindle eBooks with ease](#)

2D/3D functions

Created with the Personal Edition of HelpNDoc: [Easily create HTML Help documents](#)

cross_sections

Created with the Personal Edition of HelpNDoc: [Free Qt Help documentation generator](#)

paths2sections

Created with the Personal Edition of HelpNDoc: [Easily create eBooks](#)

path_scaling_sections

Created with the Personal Edition of HelpNDoc: [Free iPhone documentation generator](#)

bezier_surface

Created with the Personal Edition of HelpNDoc: [Free CHM Help documentation generator](#)

bezier_smooth

Created with the Personal Edition of HelpNDoc: [Free CHM Help documentation generator](#)

midpt_smooth

Created with the Personal Edition of HelpNDoc: [Easy CHM and documentation editor](#)

in_polyline

Created with the Personal Edition of HelpNDoc: [Produce electronic books easily](#)

Paths

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

arc_path

Created with the Personal Edition of HelpNDoc: [Free help authoring tool](#)

bspline_curve

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

bezier_curve

Created with the Personal Edition of HelpNDoc: [Free help authoring environment](#)

helix

Created with the Personal Edition of HelpNDoc: [Easily create EBooks](#)

golden_spiral

Created with the Personal Edition of HelpNDoc: [Qt Help documentation made easy](#)

archimedean_spiral

Created with the Personal Edition of HelpNDoc: [Easily create Help documents](#)

sphere_spiral

Created with the Personal Edition of HelpNDoc: [Easy CHM and documentation editor](#)

torus_knot

Created with the Personal Edition of HelpNDoc: [Benefits of a Help Authoring Tool](#)

Extrusion

Created with the Personal Edition of HelpNDoc: [Easily create Help documents](#)

box_extrude

Created with the Personal Edition of HelpNDoc: [Create cross-platform Qt Help files](#)

ellipse_extrude

Created with the Personal Edition of HelpNDoc: [Easy to use tool to create HTML Help files and Help web sites](#)

stereographic_extrude

Created with the Personal Edition of HelpNDoc: [Free help authoring environment](#)

rounded_extrude

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

bend_extrude

Created with the Personal Edition of HelpNDoc: [Free Qt Help documentation generator](#)

2D Shape

Created with the Personal Edition of HelpNDoc: [Easily create Help documents](#)

shape_taiwan

Created with the Personal Edition of HelpNDoc: [Produce electronic books easily](#)

shape_arc

Created with the Personal Edition of HelpNDoc: [Free PDF documentation generator](#)

shape_pie

Created with the Personal Edition of HelpNDoc: [Generate Kindle eBooks with ease](#)

shape_circle

Created with the Personal Edition of HelpNDoc: [Qt Help documentation made easy](#)

shape_ellipse

Created with the Personal Edition of HelpNDoc: [Single source CHM, PDF, DOC and HTML Help creation](#)

shape_square

Created with the Personal Edition of HelpNDoc: [Free help authoring tool](#)

shape_trapezium

Created with the Personal Edition of HelpNDoc: [Create HTML Help, DOC, PDF and print manuals from 1 single source](#)

shape_cyclicpolygon

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

shape_pentagram

Created with the Personal Edition of HelpNDoc: [Generate Kindle eBooks with ease](#)

shape_starburst

Created with the Personal Edition of HelpNDoc: [Easy EBook and documentation generator](#)

shape_superformula

Created with the Personal Edition of HelpNDoc: [Generate EPub eBooks with ease](#)

shape_glue2circles

Created with the Personal Edition of HelpNDoc: [Free Web Help generator](#)

shape_path_extend

Created with the Personal Edition of HelpNDoc: [Generate EPub eBooks with ease](#)

2D Shape extrusions

Created with the Personal Edition of HelpNDoc: [Produce electronic books easily](#)

path_extrude

Created with the Personal Edition of HelpNDoc: [Single source CHM, PDF, DOC and HTML Help creation](#)

ring_extrude

Created with the Personal Edition of HelpNDoc: [Easily create EPub books](#)

helix_extrude

Created with the Personal Edition of HelpNDoc: [Produce online help for Qt applications](#)

golden_spiral_extrude

Created with the Personal Edition of HelpNDoc: [Free HTML Help documentation generator](#)

archimedean_spiral_extrude

Created with the Personal Edition of HelpNDoc: [Free help authoring environment](#)

sphere_spiral_extrude

Created with the Personal Edition of HelpNDoc: [Free EPub producer](#)

Utils

Created with the Personal Edition of HelpNDoc: [Full-featured Help generator](#)

util/sub_str

Created with the Personal Edition of HelpNDoc: [iPhone web sites made easy](#)

New topic

Created with the Personal Edition of HelpNDoc: [Easily create EBooks](#)

New topic

Created with the Personal Edition of HelpNDoc: [Produce electronic books easily](#)

New topic

Created with the Personal Edition of HelpNDoc: [Generate EPub eBooks with ease](#)

New topic

Created with the Personal Edition of HelpNDoc: [Free EPub producer](#)

New topic

Created with the Personal Edition of HelpNDoc: [Create HTML Help, DOC, PDF and print manuals from 1 single source](#)

New topic

Created with the Personal Edition of HelpNDoc: [Free EPub producer](#)

New topic

Created with the Personal Edition of HelpNDoc: [Free Qt Help documentation generator](#)

New topic

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

New topic

Created with the Personal Edition of HelpNDoc: [Full-featured Help generator](#)

New topic

Created with the Personal Edition of HelpNDoc: [Write eBooks for the Kindle](#)

New topic

Created with the Personal Edition of HelpNDoc: [Create help files for the Qt Help Framework](#)

BOSL Library

Created with the Personal Edition of HelpNDoc: [Generate Kindle eBooks with ease](#)

Nut Job Library

Created with the Personal Edition of HelpNDoc: [Easily create Web Help sites](#)
