



Example AsciiDoc Document

Table of Contents

1. Example chapter	1
1.1. Example chapter section	1
1.1.1. Example section	1
1.1.1.1. example sub section	1
1.1.1.2. example dynamic includes section	1

Chapter 1. Example chapter

1.1. Example chapter section

With some example text. And folowed by an example sub section.

1.1.1. Example section

1.1.1.1. example sub section

With some test text here.

1.1.1.2. example dynamic includes section

Listing 1. exampleJSON.txt

```
{
  "glossary": {
    "title": "example glossary",
    "GlossDiv": {
      "title": "S",
      "GlossList": {
        "GlossEntry": {
          "ID": "SGML",
          "SortAs": "SGML",
          "GlossTerm": "Standard Generalized Markup Language",
          "Acronym": "SGML",
          "Abbrev": "ISO 8879:1986",
          "GlossDef": {
            "para": "A meta-markup language, used to create markup
languages such as DocBook.",
            "GlossSeeAlso": ["GML", "XML"]
          },
          "GlossSee": "markup"
        }
      }
    }
  }
}
```

Listing 2. *exampleinclude.txt*

This is an example raw text file
 the file contains just text and could be a log file or output from some tests
 a list of these can be compiled and automatically pulled in
 It can also be source code or JSON or siilar and can then even be highlighted by syntax.

```
:leveloffset: +2

= Description of logo
:doc-sub: true
:OLDlocaldir: {localdir}
:OLDimagesdir: {imagesdir}
:localdir: ./chapter-logo/
:imagesdir: {localdir}

== Example chapter section
With some example text.
And folowed by an example sub section.

.Example of inner logo with text as mix
image:../VSR-textndLogoGreen.png[]

.VSR logo scad file with diffreent options
//Script to create a wire mesh cube with 8x8x8 empty spaces
//Virtual Space and Global communications research department logo base object
with 8.2 cm side length
//consisting of the multiplied basic primitives of an x,y,and z axis beam
iterate in one dimension in loops
$fn=100;
module x_beam(){
    cube([82,1,1]);
}
module y_beam(){
    cube([1,82,1]);
}
module z_beam(){
    cube([1,1,82]);
}
module vsr_cube(){
    union(){
        //xbeam
        for (xj=[0:10:80]){
            for (xi=[0:10:80]){
                translate([0,xi,xj])
                x_beam();
            }
        }
    }
}
```

```

//ybeam
for (yj=[0:10:80]){
    for (yi=[0:10:80]){
        translate([yi,0,yj])
        y_beam();
    }
}
//zbeam
for (zj=[0:10:80]){
    for (zi=[0:10:80]){
        translate([zi,zj,0])
        z_beam();
    }
}
}
//basic primitive for VSR logo block.
color("#000",.25)
    vsr_cube();
// base cube with centre cube subtracted
*difference(){
    //basic primitive for VSR logo block.
    color("#000",.25)
        vsr_cube();
    //Whole to subtract
    hull() {
        translate ([30,50,70]) x_block();
        translate ([10,10,60]) x_block();
        translate ([70,40,50]) x_block();
        translate ([50,00,40]) x_block();
        translate ([20,70,30]) x_block();
        translate ([00,30,20]) x_block();
        translate ([60,60,10]) x_block();
        translate ([40,20,00]) x_block();
    }
}
//Just the Centre Block as a block
*hull() {
    translate ([30,50,70]) x_block();
    translate ([10,10,60]) x_block();
    translate ([70,40,50]) x_block();
    translate ([50,00,40]) x_block();
    translate ([20,70,30]) x_block();
    translate ([00,30,20]) x_block();
    translate ([60,60,10]) x_block();
    translate ([40,20,00]) x_block();
}
module x_block(){
    // Cubes as corners for complex wire frame centre
    *color([0,1,0]) translate ([02,02,02]) cube([8,8,8]);
}

```

```

//spheres as corners for smoothe wireframe centre
translate ([6,6,6]) sphere (r=1);
}
//Just 8 centre points as corners
union () {
  //1st set
  hull() {
    translate ([30,50,70]) x_block();
    translate ([10,10,60]) x_block();
  }
  hull() {
    translate ([70,40,50]) x_block();
    translate ([50,00,40]) x_block();
  }
  hull() {
    translate ([20,70,30]) x_block();
    translate ([00,30,20]) x_block();
  }
  hull() {
    translate ([60,60,10]) x_block();
    translate ([40,20,00]) x_block();
  }
  //second set
  hull() {
    translate ([30,50,70]) x_block();
    translate ([20,70,30]) x_block();
  }
  hull() {
    translate ([10,10,60]) x_block();
    translate ([00,30,20]) x_block();
  }
  hull() {
    translate ([70,40,50]) x_block();
    translate ([60,60,10]) x_block();
  }
  hull() {
    translate ([50,00,40]) x_block();
    translate ([40,20,00]) x_block();
  }
  //third set
  hull() {
    translate ([30,50,70]) x_block();
    translate ([70,40,50]) x_block();
  }
  hull() {
    translate ([10,10,60]) x_block();
    translate ([50,00,40]) x_block();
  }
  hull() {
    translate ([20,70,30]) x_block();

```

```
        translate ([60,60,10]) x_block();
    }
    hull() {
        translate ([00,30,20]) x_block();
        translate ([40,20,00]) x_block();
    }
}

:leveloffset!:
```