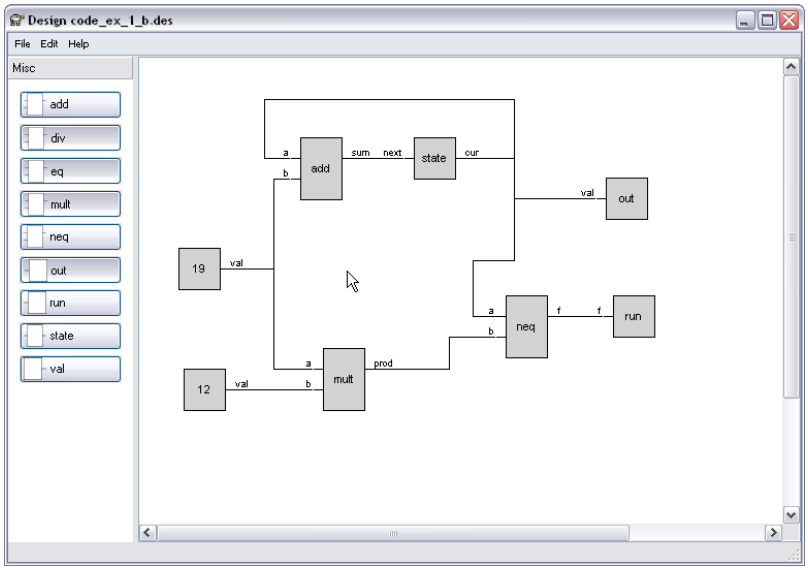
Memention Designer / Short Manual

Getting familiar

The Memention Designer window has a toolkit to the left and a design canvas to the right.



The toolkit contain the objects that can be used to build a design. The objects can be organized into different categorizes accessible through tabs (not shown in demo).

The objects on the design canvas can be selected, moved and deleted as expected with the mouse. The Memention Designer keeps the last 20 changes for undos.

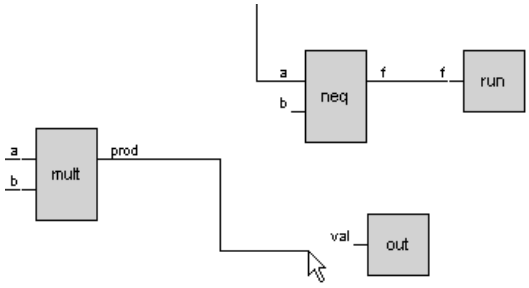
The output of the design is generated by selecting Generate from the Edit menu.

Click-drag object

An object is added to a design by clicking on the object in the toolkit and dragged into the design view.

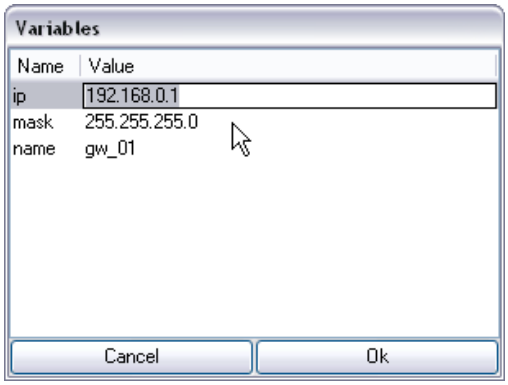
Click on connector

Connecting objects is done by clicking on a connector and then clicking on the ending connector. The path can be laid out by clicking on the canvas. Each click will add a corner to the connection path. A press on Backspace will delete the last placed corner and a press on Escape will end and remove the connection path being created.



Double-click on object

Objects can contain editable variables. A variables window can be activated by double-clicking on an object and in this window can these variables be edited.

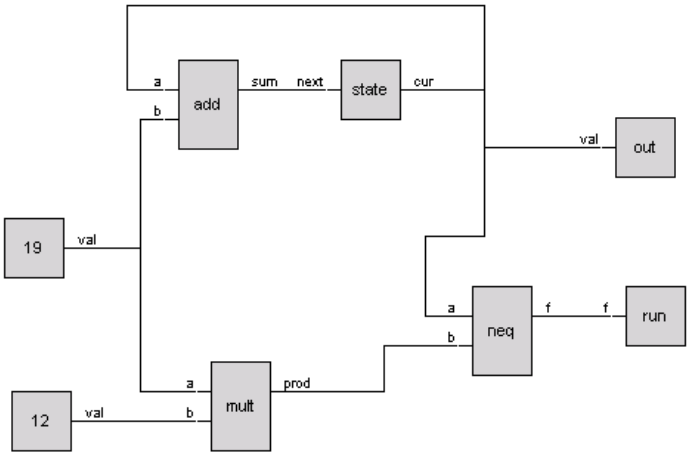


Right-click-move

Moving around on the design view can easily be done by right-click-move to pan/move.

Example code_ex_1_b.des

The code_ex_1_b.des will generate a small and simple C program that will output a list of values each added by 19 of the previous value. The toolkit contains a set of simple operators and objects to output and end the running demo loop.

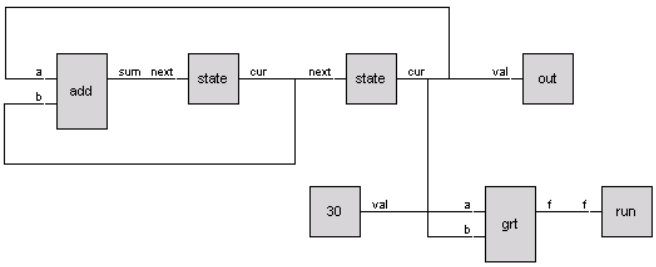


The demo will generate the following code. Try double clicking on the objects “19” and “12” and alter the “name” which also act as their values.

```
/*----------------------------------------------------------------
 * Generated with Memention Designer, Code Example
 *----------------------------------------------------------------*/

#include <stdio.h>

/* there are 8 blocks */
int running;
/* there are 1 out blocks */
int state_curr_1;
int state_next_1;

int main(int argc, char *argv[]) {
  running = 1;
  state_curr_1 = 0;
  while (running) {
    state_next_1 = (state_curr_1 + 19);
    running = (state_curr_1 != (19 * 12));
    state_curr_1 = state_next_1;
    printf("out = %d\n", state_curr_1);
  }
  return 0;
}
```

Example code_ex_2_b.des

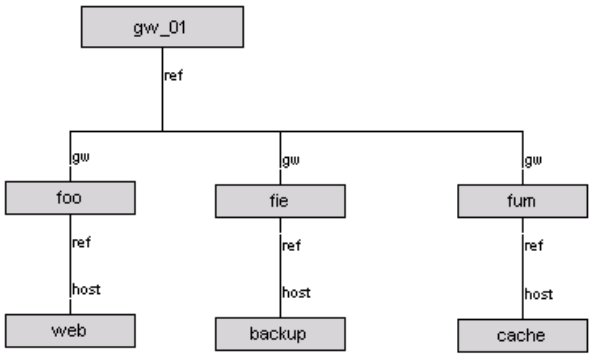
The example code_ex_2_b.des will also as the above example generate a small and simple C program. The program will generate a couple of numbers from the famous Fibonacci series.



```
/*-----------------------------------------------------------------
 * Generated with Memention Designer, Code Example
 *-----------------------------------------------------------------*/

#include <stdio.h>

/* there are 7 blocks */
int running;
/* there are 1 out blocks */
int state_curr_1;
int state_next_1;
int state_curr_2;
int state_next_2;

int main(int argc, char *argv[]) {
  running = 1;
  state_curr_1 = 1;
  state_curr_2 = 0;
  while (running) {
    state_next_1 = (state_curr_2 + state_curr_1);
    state_next_2 = state_curr_1;
    running = (30 > state_curr_2);
    state_curr_1 = state_next_1;
    state_curr_2 = state_next_2;
    printf("out = %d\n", state_curr_2);
  }
  return 0;
}
```

Example section_cfg_ex_b.des

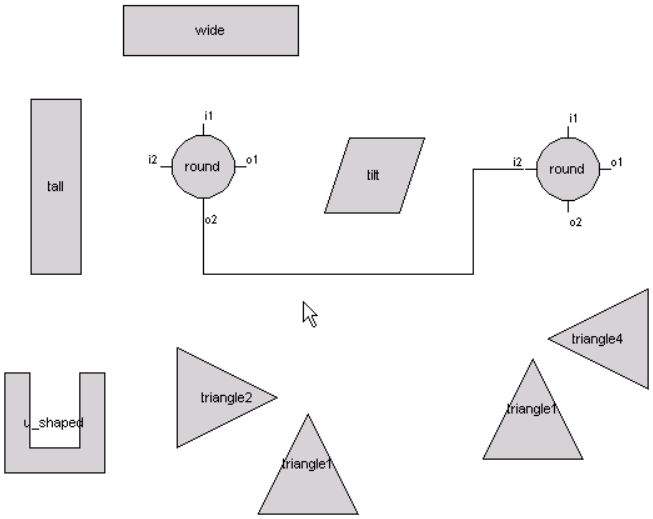
The example section_cfg_ex_b.des will output a configuration with a fictitious file format.



```
#
# Generated with Memention Designer, Section Config Example
#

.include gateway_pkg
.include host_pkg
.include service_pkg

[gateway gw_01]
addr = 192.168.0.1
name = gw_01

[host foo]
addr = 192.168.0.101
name = foo
gateway = 192.168.0.1

[host fie]
addr = 192.168.0.102
name = fie
gateway = 192.168.0.1

[host fum]
addr = 192.168.0.103
name = fum
gateway = 192.168.0.1

[service web]
enable = true
debug = 1
log = 1
port = 80
host = foo
pidfile = /var/opt/web.pid

[service backup]
enable = true
debug = 1
log = 1
port = 9000
host = fie
pidfile = /var/opt/backup.pid

[service cache]
enable = true
debug = 1
log = 1
port = 8181
host = fum
pidfile = /var/opt/cache.pid
```

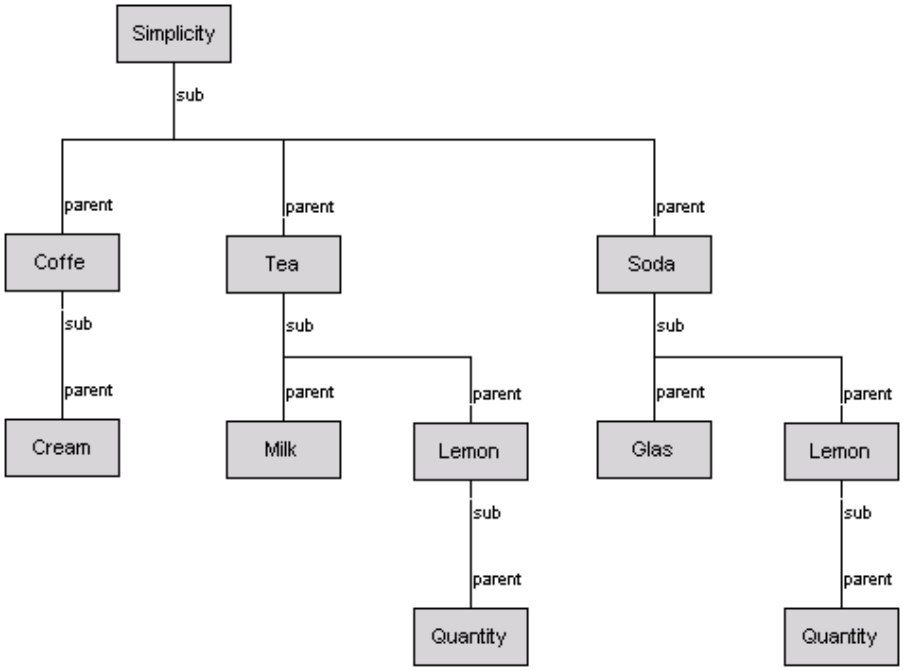
Example shapes_ex_1_b.des

The example shapes_ex_1_b.des purpose is to show a couple of possible object shapes. Object shapes are defined as a set of vector instructions similar to turtle graphics commands.



Example xml_ex_1_b.des

The example xml_ex_1_b.des will output a XML hierarchy. Double clicking on objects will enable editing of tag names and leaf content.



```
<xml>
  <Simplicity>
    <Coffe>
      <Cream>No</Cream>
    </Coffe>
    <Tea>
      <Milk>Yes</Milk>
      <Lemon>
        <Quantity>0.5</Quantity>
      </Lemon>
    </Tea>
    <Soda>
      <Glas>Tall</Glas>
      <Lemon>
        <Quantity>0.62</Quantity>
      </Lemon>
    </Soda>
  </Simplicity>
</xml>
```