A possible way for handling a set (quasi group) of objects according to their properties

Playing with object properties

Working with GeoGebra (**Mac OS X 10.6.8**, GeoGebra: **4.4.23**) I've faced with a problem many times, to change a property (e.g. color) of certain objects, which fulfill some criterion, (e.g. opacity is equal to 1), or e.g. maintain a checkbox, which controls objects. Since there is no group feature in this GeoGebra version, this work shows a possible solution of the problem.
The basis of this solution is, that every *.ggb* file (GeoGebra applet) contains a *geogebra.xml* file which stores all the necessary information about object properties.

Table of Contents

1. THE TASK IN BRIEF

- create a list of object names which fulfill the appropriate criterion,
- create a JavaScript or GeoGebra Script, which performs the necessary action,
- insert the list of object names into the JavaScript or GeoGebra Script,
- insert the JavaScript or GeoGebra Script into the applet,
- execute the JavaScript or GeoGebra Script.

2. CREATE A LIST OF OBJECT NAMES

Necessary tools: terminal window, text editor.
This task is performed in two phases:

- extract all the object names and their property (which belongs to the criterion, e.g. opacity) from the *element* blocks of the *geogebra.xml* file, with *gxx.sh*,
- filter those object names which fulfill the criterion (e.g. opacity is equal to 1) from the output of *gxx.sh* (*gg_property-by_parameter-list.txt*) to create a list of object names.

A rough schema is: *gxx.sh* | filter

2.1 Extract object names and their property

2.1.1. gxx.sh

See: #Shell script to copy & paste

2.1.2. Install and configure gxx.sh

- Copy & paste *gxx.sh* to your system and save it,
- Set *dir-1, dir-2, dir-3* according to your environment,

```
## BEGIN: SET dir-1, dir-2, dir-3 ##
DIR=/dir-1
ELEMENT_PROPERTY_FILE=$DIR/dir-2/gg_element-property.txt
OUTPUT_FILE=$DIR/dir-3/gg_property-by_parameter-list.txt
## END: SET dir-1, dir-2, dir-3 ##
```

  dir-1, dir-2 and dir-3 don't have to be different.
- Set it to executable: `chmod 744 gxx.sh`

2.1.3. Execute gxx.sh

- **a.** Command ./*gxx.sh* directory_of_GeoGebra_file_you_need
- **b.** Choose the name of the GeoGebra file you need
- **c.** Choose the object property you need
- **d.** Output *gg_property-by_parameter-list.txt* (you had to set *OUTPUT_FILE*)

Example

```
$ ./gxx.sh /Volumes/work/GG/        # a.

a-1.ggb  a-2.ggb  a.ggb  gxx_demo-2.gg  gxx_demo.ggb

ggb-file?
```

```
gxx_demo-2.ggb                                # b.
Archive:  /Volumes/work/GG//gxx_demo-2.ggb
  inflating: geogebra.xml
animation
auxiliary
bgColor
caption
eigenvectors
eqnStyle
font
javascript
labelMode
labelOffset
layer
lineStyle
matrix
objColor
show
startPoint

property?                                     # c.
lineStyle
$
```

2.2 Filter object names fulfilling the criterion

The goal is a list of object names, which comes from filtering the output of *gxx.sh* (*gg_property-by_parameter-list.txt*).
E.g.:

```
"a109","a117","a142","a145","a160","a163","a189","a20","a209","a22
","a223","a228","a269","a286","a369","a391","a399","a419","a477","
a484","a516","a523","a558","a561","a612","a613","a625","a647","a65
7","a664","a665","a682","a688","a754","a760","a761","a779","a780",
"a782","a82","a827","a836","a85","a885","a906","a922","a928","a956
","a973"
```

The filter depends on the property, what you chose when executed *gxx.sh*.
E.g.:

2.2.1 if property is „lineStyle" and value is „8"

Excerpt from *gg_property-by_parameter-list.txt* file:

```
type="conic" label="a101"    lineStyle thickness="2" type="0"
typeHidden="1"
type="conic" label="a102"    lineStyle thickness="8" type="0"
typeHidden="1"
type="conic" label="a103"    lineStyle thickness="2" type="0"
typeHidden="1"
```

- Filter:

```
awk '$4 ~ /'8'/ {print $2}' gg_property-by_parameter-list.txt |
cut -c 7- | tr '\n' ','
```

2.2.2 if property is „condition showObject”

Excerpt from *gg_property-by_parameter-list.txt* file:

```
type="conic" label="a106"   condition showObject="ShowHide"
type="conic" label="a148"   condition showObject="ShowHide"
type="conic" label="a149"   condition showObject="ShowHide"
```

- Filter:

```
cat gg_property-by_parameter-list.txt | cut -c 20-25 | tr '\n' ','
| sed 's/"//g'
```

2.2.3 if property is „objColor” and value is „1.0”

Excerpt from *gg_property-by_parameter-list.txt* file:

```
type="conic" label="a116"   objColor r="0" g="0" b="0" alpha="0.0"
type="conic" label="a117"   objColor r="0" g="0" b="0" alpha="1.0"
type="conic" label="a118"   objColor r="0" g="0" b="0" alpha="0.0"
```

- Filter:

```
awk '$7 ~ /'1.0'/ {print $2}' gg_property-by_parameter-list.txt |
cut -c 7- | tr '\n' ','
```

3. PREPARE JAVASCRIPT OR GEOGEBRA SCRIPT

Main steps:

- create a JavaScript or GeoGebra Script which performs the appropriate action (e.g. change color) on those objects, which fulfill the appropriate criterion,
- create a button in the *.ggb* file, and insert the JavaScript or GeoGebra Script

3.1 Create a JavaScript or GeoGebra Script

Necessary tool: text editor.
These scripts contain two main parts:

- a set of objects, on which an action have to be done (the list of object names which fulfill the appropriate criterion):
  - JavaScript: the value of the “Objects2Modify” variable,
  - GeoGebra Script: the list part of the “Checkbox” command,
- code, which describes the action.

3.1.1 JavaScript (e.g. a modify color action)

```
var Objects2Modify =
["a127","a15","a157","a168","a171","a196","a212","a225","a244","a264","a272","a2
75","a288","a294","a314","a350","a379","a395","a412","a467","a485","a499","a542"
,"a550","a551","a566","a575","a596","a61","a62","a651","a74","a742","a744","a808
","a815","a82","a833","a837","a839","a862","a894","a909","a912","a914","a916","a
951","a959","a960","a994"];

var i;
for (i in Objects2Modify)
    {
    ggbApplet.evalCommand("SetColor[" + Objects2Modify[i] + ", 0.25, 0.5, 1]");
    }
```

3.1.2 GeoGebra Script (e.g. a modify checkbox action)

```
Execute[{"Delete[ShowHide]"}]
Execute[{"ShowHide = Checkbox[{a340,a734,a742,a758,a765,a784,a813,a833,rso}]"}]
Execute[{"SetCoords[ShowHide, 510, 100]"}]
```

3.2 Insert the JavaScript or GeoGebra Script into the applet

Necessary tool: GeoGebra.
There are 3 buttons for this purpose in the test applet, button8 - „modify color”, button9 - „move objects” and button10 - „modify checkbox”.

- open *gxx_demo.ggb* / the appropriate one from the 3 buttons / Object Properties / Scripting / On Click,
- choose „JavaScript”, or „GeoGebra Script”, and paste the appropriate script,
- click OK, then close Object Properties / Scripting of the appropriate button.

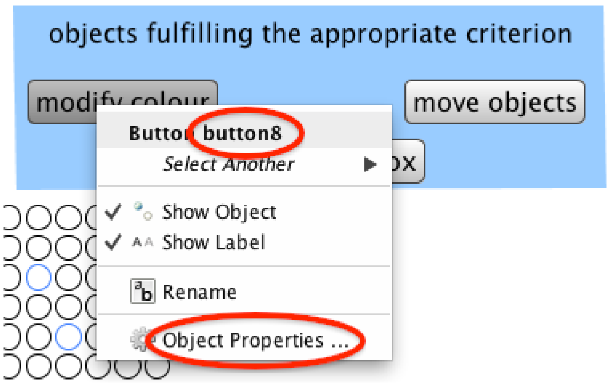
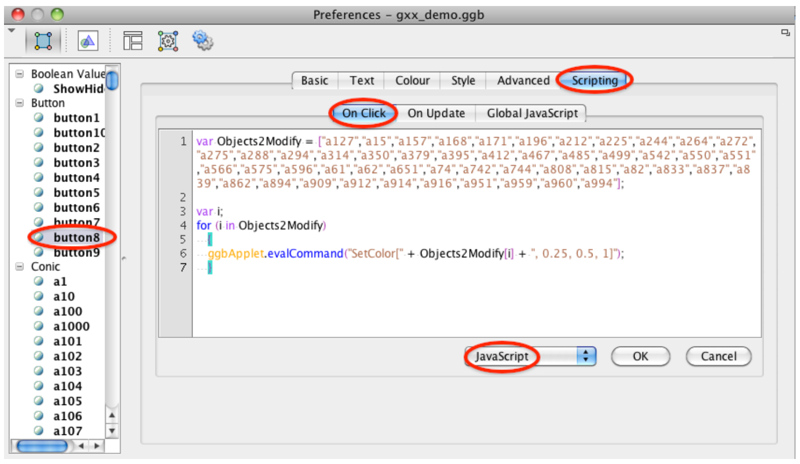


Figure 1



```
var Objects2Modify = ["a127","a15","a157","a168","a171","a196","a212","a225","a244","a264","a272",
"a275","a288","a294","a314","a350","a379","a395","a412","a467","a485","a499","a542","a550","a551"
,"a566","a575","a596","a61","a62","a651","a74","a742","a744","a808","a815","a82","a833","a837","a8
39","a862","a894","a909","a912","a914","a916","a951","a959","a960","a994"];

var i;
for (i in Objects2Modify)
  {
  ggbApplet.evalCommand("SetColor[" + Objects2Modify[i] + ", 0.25, 0.5, 1]");
  }
```

Figure 2

4. EXECUTE THE JAVASCRIPT OR GEOGEBRA SCRIPT

4.1 The test applet (gxx_demo.ggb)

The purpose of the test applet is to

- create a random selected set of objects (with about 50 elements) by means of changing a property of them (e.g.: “lineStyle thickness”), in order to let *gxx.sh* and a filter find them,
- modify another property (e.g.: „objColor”) of the set of objects above by the help of a JavaScript, or a GeoGebra Script, utilizing the output of the filter.

Description of the screen:

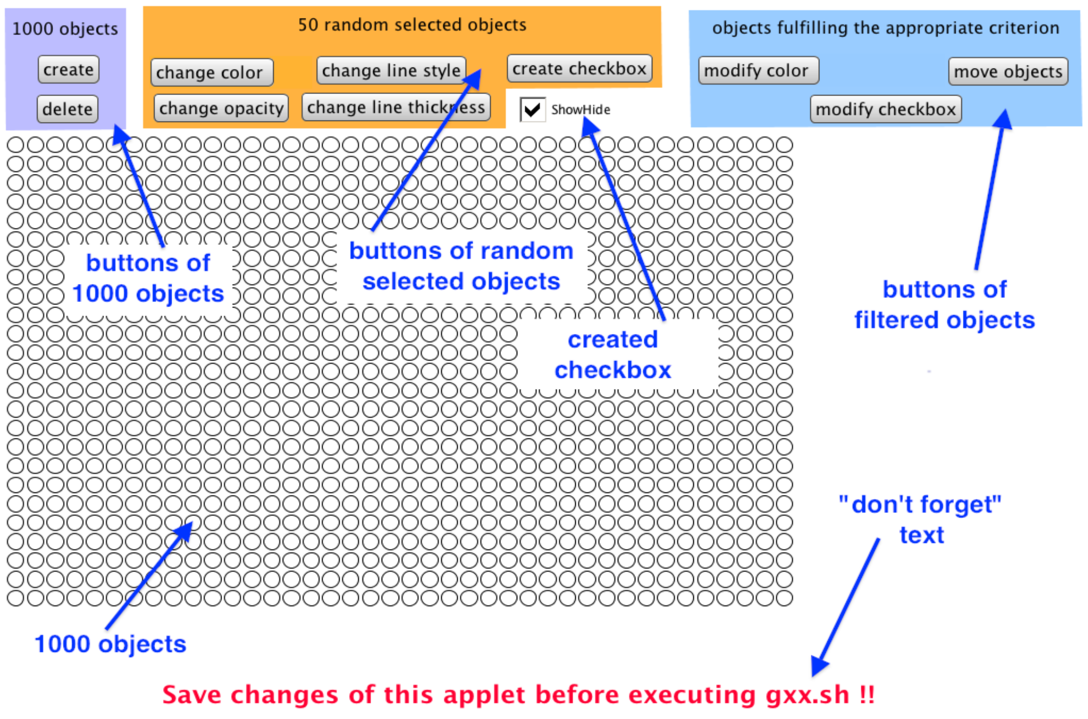


Figure 3

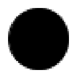
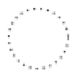
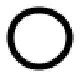
4.2 Description of buttons

4.2.1 buttons of 1000 objects

- create
  Creates 1000 pieces of new objects.
- delete
  Deletes 1000 pieces of objects and (if exists) the ShowHide checkbox.

4.2.2 buttons of random selected objects

They create a random selected set of objects from the 1000 ones by means of changing a property of about 50 objects. “Create checkbox” button creates a checkbox too.

Button	Changes a property	Figure	
change color	**from** `objColor r="0"   g="0"   b="0"  alpha="0.0"` **to** `objColor r="`**`255`**`" g="`**`127`**`" b="`**`63`**`" alpha="0.0"`		4
change opacity („alpha”)	**from** `objColor r="0" g="0" b="0" alpha="0.0"` **to** `objColor r="0" g="0" b="0" alpha="`**`1.0`**`"`		5
change line stile	**from** `lineStyle thickness="2" type="0" typeHidden="1"` **to** `lineStyle thickness="2" type="`**`20`**`" typeHidden="1"`		6
change line thickness	**from** `lineStyle thickness="2" type="0" typeHidden="1"` **to** `lineStyle thickness="`**`8`**`" type="0" typeHidden="1"`		7
create checkbox	creates a checkbox: ShowHide creates `condition showObject` property for about 50 random selected objects and sets it to: **`condition showObject="ShowHide"`**		

4.2.3 buttons of filtered objects

They modify a property of that set of objects, which was generated by “buttons of random selected objects” and was filtered from the output of *gxx.sh* (*gg_property-by_parameter-list.txt*).

- modify color
  Modifies „objColor” property of the object set to blue.
  `objColor r="63" g="127" b="255" alpha="0.0"`

  See: #setcolor
  `SetColor[" + Objects2Modify[i] + ", 0.25, 0.5, 1]`
  255 * 0.25 = 63,75;   255 * 0.50 = 127,5;   255 * 1 = 255

- move objects
  Recreates the set of objects, with modified coordinates.
- modify checkbox
  Modify the set of objects which are controlled by ShowHide checkbox.

5. EXERCISES

The 1st steps for all the exercises:

- download (once) and open *gxx_demo.ggb* in a GeoGebra window,
- tabula rasa
  - click „delete objects” button to clear old objects,
  - click „create objects” button to create new objects.

5.1. modify color

The task is to modify color of those objects to blue, where “lineStyle thickness=8”.

- Click „change line thickness” button to create a random selected set of objects with changing their “lineStyle thickness” property from 2 to 8.

  The product is like this, depending on the random selection:

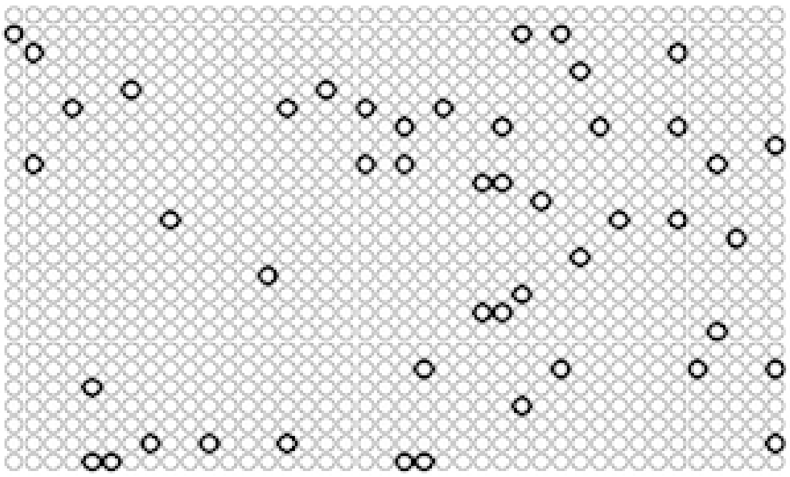


Figure 8

- save *gxx_demo.ggb*,
- execute *gxx.sh* (see: [#execute gxxsh]) to extract the „lineStyle” property from the *geogebra.xml* file,
  - `ggb-file?`
    `gxx_demo.ggb`
  - `property?`
    `lineStyle`

- filter the output of *gxx.sh* (*gg_property-by_parameter-list.txt*) to get object names fulfilling the criterion (“lineStyle thickness=8”),

```
awk '$4 ~ /'8'/ {print $2}' gg_property-by_parameter-list.txt
| cut -c 7- | tr '\n' ','
```

- copy the list generated above, it will be the value of „Objects2Modify” variable in the JavaScript (separator is comma, you don't need the last one),
- open *gxx_demo.ggb* / „modify color” button / Object Properties / Scripting / On Click
- choose „JavaScript”, and paste the following script:

```
var Objects2Modify =
["a147","a165","a21","a22","a222","a229","a236","a240","a317"
,"a345","a346","a387","a414","a470","a48","a5","a51","a518","
a529","a55","a552","a555","a588","a6","a625","a626","a642","a
659","a661","a677","a720","a741","a746","a751","a755","a764",
"a775","a779","a783","a80","a807","a817","a870","a882","a915"
,"a921","a947","a949"];

var i;
for (i in Objects2Modify)
   {
   ggbApplet.evalCommand("SetColor[" + Objects2Modify[i] + ",
0.25, 0.5, 1]");
   }
```

The value of „Objects2Modify" variable will differ after every clicking on the "buttons of random selected objects", because of their random working.

- Click OK, then close Object Properties / Scripting of the „modify color" button,
- click „modify color" button to execute JavaScript.

The product is like this, compare it with Figure 8:

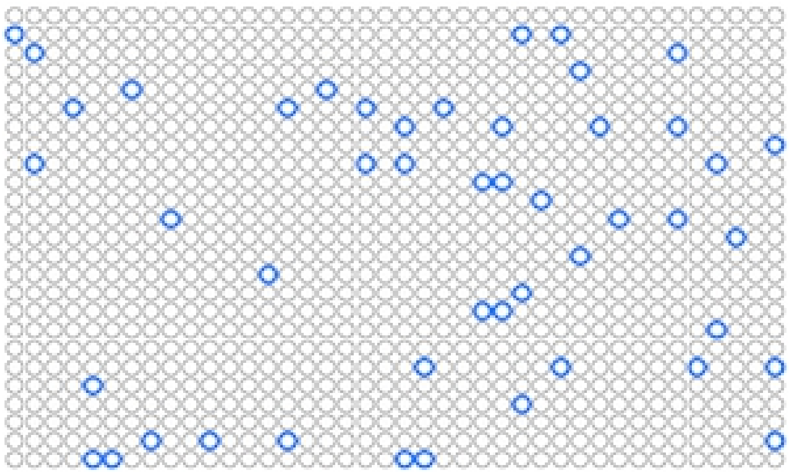


Figure 9

5.2. maintain checkbox

The task is to modify the set of those objects which are controlled by „ShowHide" checkbox. The modification is to delete 10 objects from the most upper rows, and add „rso" object. „rso" object is the light brown hexagon which frames „buttons of random selected objects".

- Click „create checkbox" button to create checkbox „ShowHide", and a random selected set of objects which are controlled by „ShowHide" checkbox.

The product is like this, depending on the random selection:

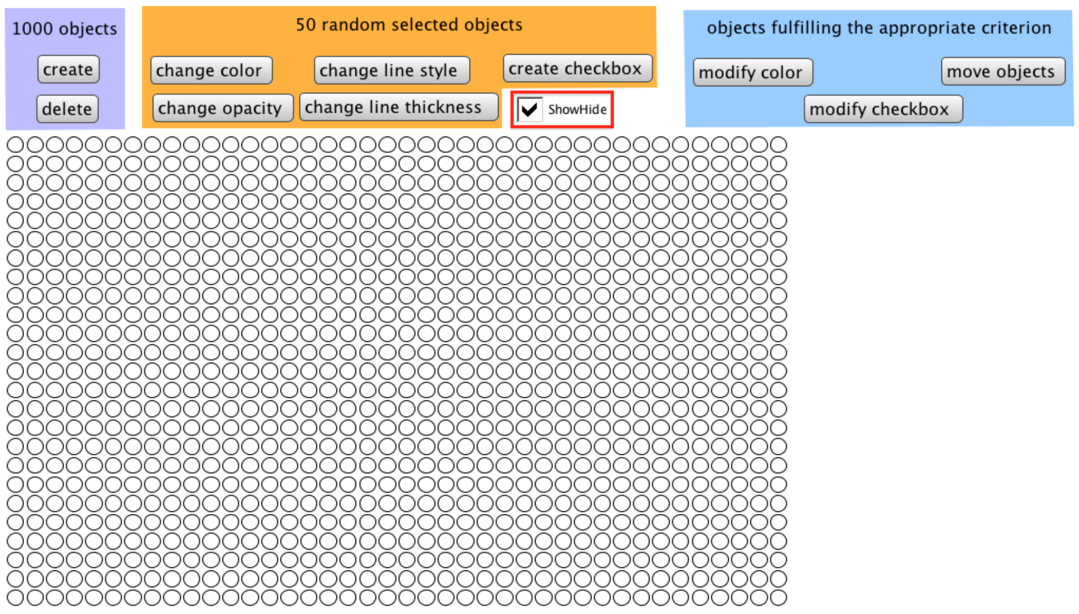


Figure 10

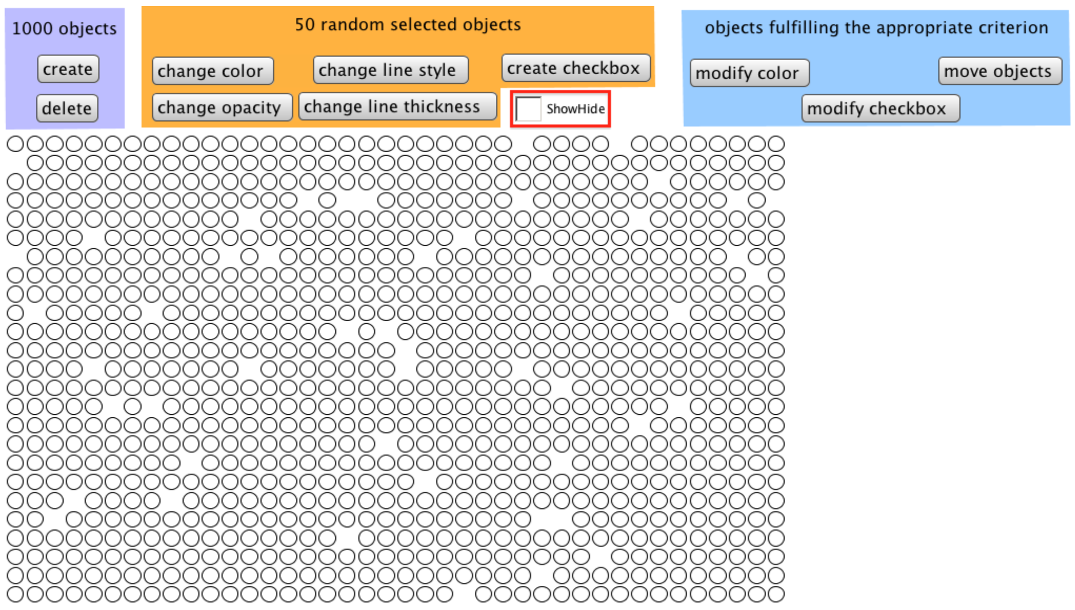


Figure 11

- save *gxx_demo.ggb*,
- execute *gxx.sh* (see: #execute gxxsh) to extract „Condition” property from *geogebra.xml* file,
  - `ggb-file?`
    `gxx_demo.ggb`
  - `property?`
    `Condition`
- filter the output of *gxx.sh* (*gg_property-by_parameter-list.txt*) to get object names fulfilling the criterion (beeing controlled by „ShowHide” checkbox).

```
cat gg_property-by_parameter-list.txt | cut -c 20-25 | tr
'\n' ',' | sed 's/"//g'
```

The product is like this:

```
a111,a138,a163,a188,a189,a204,a209,a24
,a262,a290,a309,a340,a393,a406,a408,a435,a469,a485,a493,a501,a507,
a541,a578,a580,a602,a608,a635,a68
,a708,a719,a721,a732,a734,a742,a758,a765,a784,a813,a833,a856,a858,
a859,a867,a878,a880,a914,a921,a987,a992
```

- 10 objects from the most upper rows are to delete:

```
a856,a858,a859,a867,a878,a880,a914,a921,a987,a992
```

- 1 object is to add:

```
rso
```

- The new set of objects being controlled by „ShowHide” checkbox is:

```
a111,a138,a163,a188,a189,a204,a209,a24
,a262,a290,a309,a340,a393,a406,a408,a435,a469,a485,a493,a501,a507,
a541,a578,a580,a602,a608,a635,a68
,a708,a719,a721,a732,a734,a742,a758,a765,a784,a813,a833,rso
```

- copy the list generated above, it will be the list part of the „Checkbox” command in the GeoGebra Script (separator is comma, you don't need the last one),
- open *gxx_demo.ggb* / „modify checkbox” button / Object Properties / Scripting / On Click
- choose „GeoGebra Script”, and paste the following script:

```
Execute[{"Delete[ShowHide]"}]
Execute[{"ShowHide =
Checkbox[{a111,a138,a163,a188,a189,a204,a209,a24
,a262,a290,a309,a340,a393,a406,a408,a435,a469,a485,a493,a501,
a507,a541,a578,a580,a602,a608,a635,a68
,a708,a719,a721,a732,a734,a742,a758,a765,a784,a813,a833,rso}]
"}]
Execute[{"SetCoords[ShowHide, 510, 100]"}]
```

  The list part of the “Checkbox” command will differ after every clicking on the "buttons of random selected objects", because of their random working.

- Click OK, then close Object Properties / Scripting of the „modify checkbox” button,
- click „modify checkbox” button to execute GeoGebra Script,
- click „ShowHide” checkbox to visualize the new set of objects being controlled by „ShowHide” checkbox.

  The product is like this, compare it with Figure 11 (look at the most upper 4 rows and the lack of the light brown hexagon which framed „buttons of random selected objects”):

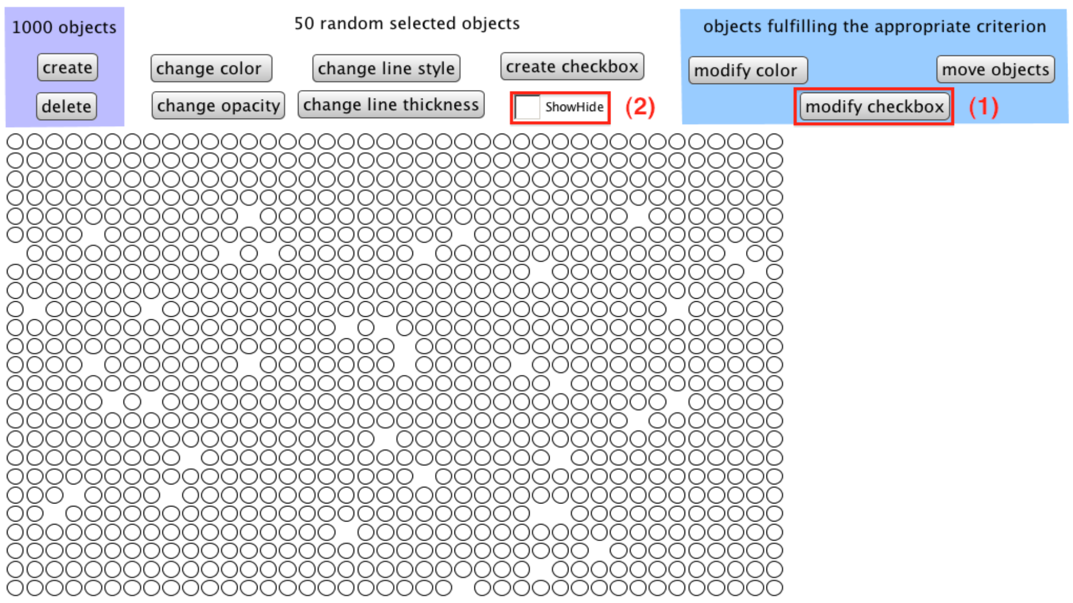


Figure 12

5.3. move objects

The task is to move those objects, where opacity = 1 (with other words: alpha="1.0")

- Click „change opacity" button to create a random selected set of objects with changing their „alpha" property from 0.0 to 1.0.

  The product is like this, depending on the random selection:

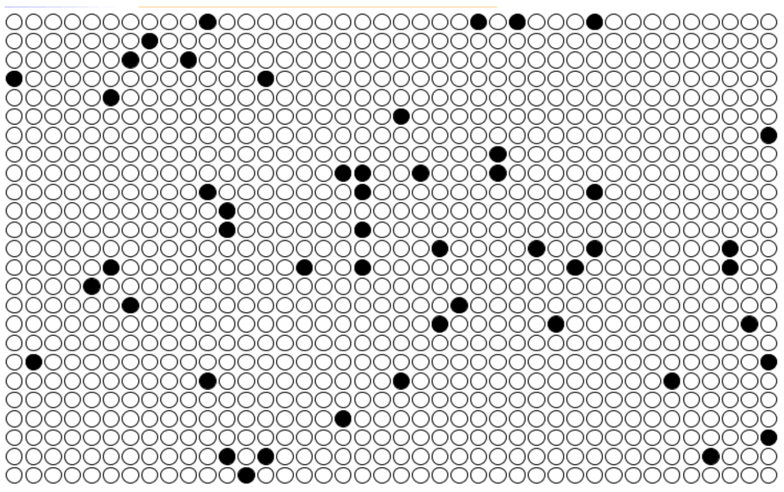


Figure 13

- save *gxx_demo.ggb*,
- execute *gxx.sh* (see: [#execute gxxsh](#execute-gxxsh)) to extract the „objColor" property from the *geogebra.xml* file,

  -
    ```
    ggb-file?
    gxx_demo.ggb
    ```

  -
    ```
    property?
    objColor
    ```

- filter the output of *gxx.sh* (*gg_property-by_parameter-list.txt*) to get object names fulfilling the criterion ('alpha="1.0"'),
  `awk '$7 ~ /'1.0'/ {print $2}' gg_property-by_parameter-list.txt | cut -c 7- | tr '\n' ','`
- copy the list generated above, it will be the value of „Objects2Modify” variable in the JavaScript (separator is comma, you don't need the last one),
- open *gxx_demo.ggb* / „move objects” button / Object Properties / Scripting / On Click
- choose „JavaScript”, and paste the following script:

```
var Objects2Modify =
["a120","a13","a138","a211","a221","a235","a242","a280","a343","a349","a359","a367","a384","a405","a
446","a456","a459","a470","a478","a503","a508","a511","a518","a52","a532","a539","a54","a572","a611"
,"a619","a631","a658","a659","a662","a666","a706","a760","a77","a781","a806","a841","a854","a887","a
890","a928","a971","a985","a987","a991"];

var i;
var j = 0;
var k = 2;
for (i in Objects2Modify)
   {
   j ++;
   ggbApplet.evalCommand(Objects2Modify[i] + "= Circle[(" + j * 1.2 + ", " + -k + "),  0.5]");
   if (j == 40)
      {
      j = 0;
      k+=1.2;
      }
   }
```

The value of „Objects2Modify” variable will differ after every clicking on the "buttons of random selected objects", because of their random working.

- Click OK, then close Object Properties / Scripting of the „move objects” button,
- click „move objects” button to execute JavaScript.

The product is like this, compare it with Figure 13:

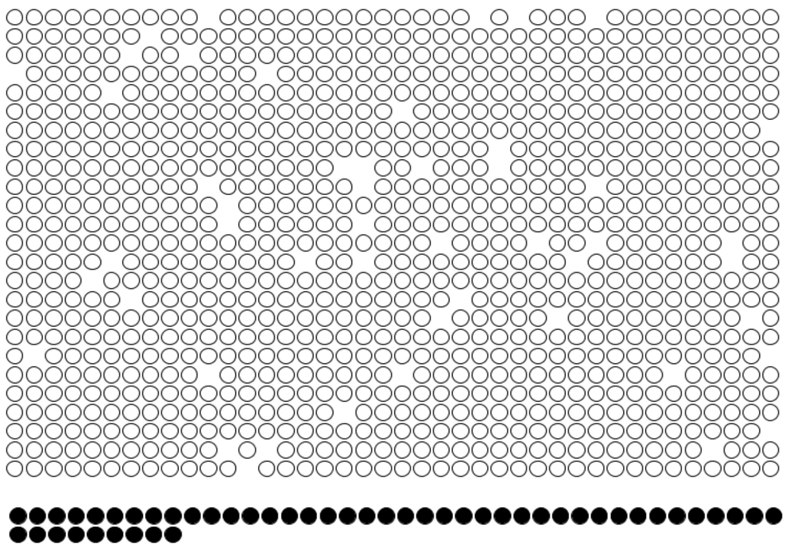


Figure 14

6. APPENDIX

Shell script to copy & paste

```
#!/bin/ksh

#****************************** gxx.sh ***************

# gxx - Geogebra Xml eXtractor

if test $# -lt 1
then
        echo "Usage:   $0 ggb-directory"
        exit 1
fi

GGB_DIR=$1
XML_FILE=geogebra.xml

## BEGIN: SET dir-1, dir-2, dir-3 ##
DIR=/dir-1
ELEMENT_PROPERTY_FILE=$DIR/dir-2/gg_element-property.txt
OUTPUT_FILE=$DIR/dir-3/gg_property-by_parameter-list.txt
## END: SET dir-1, dir-2, dir-3 ##

echo
cd $GGB_DIR
ls *.ggb
echo
echo ggb-file?
read GGB_FILE
cd $DIR

unzip $GGB_DIR/$GGB_FILE $XML_FILE

cat $XML_FILE                                                                  |\

############################ extract elements and their properties ############################
awk 'substr($0, 1, 8) == "<element"                    {print $0;
                                                        start = "yes";}
     start == "yes" && substr($0, 1, 1) == "\t"         {print $0;}
     substr($0, 1, 9) == "</element"                    {start = "no";}'       |\

sed 's/[<>/]//g' > $ELEMENT_PROPERTY_FILE

############################ show unique property list ############################
cat $ELEMENT_PROPERTY_FILE                                                     |\
awk 'substr($0, 1, 1) == "\t" {print $0;}'                                     |\
tr -d '\t'                                                                     |\
cut -d ' ' -f 1                                                                |\
sort                                                                           |\
sort -u

echo
echo

############################ read parameter ############################
echo "property?"
read PP

echo "element type=" > $DIR/property/pp_file
echo $PP >> $DIR/property/pp_file

############################ create property list by parameter ############################
cat $ELEMENT_PROPERTY_FILE                                                     |\
egrep -f $DIR/property/pp_file                                                 |\

awk 'substr($0, 1, 7) == "element"    {possible_printable_element = substr($0, 9);}
     $1 == pp                         {printf("\n%-35s\t\t", possible_printable_element);
                                       print $0;
                                       possible_printable_element = "";}' pp=$PP    |\

grep -v '^$'                                                                   |\
sort -k 1 -k 2 > $OUTPUT_FILE

rm $XML_FILE $ELEMENT_PROPERTY_FILE $DIR/property/pp_file
```