SDIA
a system for SD diagrams typesetting

Version: 1.0 Date: Jun 29 2005
You can get the newest version at **http://www.square.cz/sdia/**

Contents

1. Preface

This system was designed for quick making SD diagrams in Encapsulated PostScript (.eps) format usable at all operating systems with no special software needed. All you need is the (ASCII) text editor, even the most ordinary one (like vi on UNIX or Notepad on Windows). Of course you don't need to know anything about PostScript, but it could be your advantage if you want to modify the system to your special needs.

Encapsulated PostScript diagrams can be used for many typesetting programs directly: TEX (with dvips), Adobe PageMaker, Quark X-press and others. Converting PostScript to another format is not a task solved by this system. However this could be done with free programs also, for example ghostscript.

The system is not WYSIWYG – it can't be because it should be simple and usable at computers with no special SW. But maybe you will need some helper application at the beginning of learning. There is one. Open sdia_generator.html file in your HTML browser (JavaScript and CSS capable). It will help you to create your first sdiagrams and to understand how the system works. You don't need nor network connection nor any other file.

It's really easy!

2. Usage

2.1. Basic idea

There is a "master" file sdia.eps which you should copy for each new diagram you want to make. Then you need to edit 2 parts in the new copy. The user data section contains all the diagram information such as dancers. This is described in the following chapters. Now we talk about the first editable part, the BoundingBox.

BoundingBox says the size (measured in PostScript points) of the result picture to your typesetting system. Four numbers mean: left, bottom, right and top border. As you can see this is where some computing is needed, but very simple. In most cases the following heuristics works:

*%%BoundingBox: 0 0 23+X*23 23+Y*23*

X, Y: coordinates
maximal horizontal, resp. vertical dancer coordinates used at the picture

2.2. MainStream usage

User data contains one line for each dancer or standalone command. Standalone commands are described in Chapter Challenge commands, format of dancer line is:

⟨*dancer label*⟩ [⟨*parameters*⟩] ⟨*direction*⟩ ⟨*coordinates*⟩ ⟨*gender*⟩

dancer label: one of letters A-H or text (enclosed in parenthesis)
Letters A-H are replaced by graphical sign at output. Text label allows to sign dancers with real letters or numbers. Empty text label () is used to draw a dancer with no visible label.

parameters: not needed
used for special effects described later

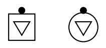
direction: one of e, w, n, s, v, h or x
east, west, north, south, (both) vertical, (both) horizontal, any

coordinates: two numbers: x and y.
Left bottom corner is 0 0, step is choosen as one "dancer position".

gender: g or b
girl or boy

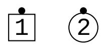
Example 1: Normal couple facing north

```
A n 0 0 b
A n 1 0 g
```



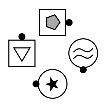
Example 2: Same couple with dancers labeled "1" and "2"

```
(1) n 0 0 b
(2) n 1 0 g
```



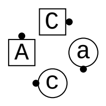
Example 3: Right-hand star, dancers labeled with graphical signs

```
A n 0   0.5 b
B s 1   0.5 g
C e 0.5 1   b
D w 0.5 0   g
```



Example 4: Right-hand star with dancers labeled "A", "a", "C", "c"

```
(A) n 0   0.5 b
(a) s 1   0.5 g
(C) e 0.5 1   b
(c) w 0.5 0   g
```



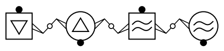
2.3. Plus hands

You can use LH or RH parameter to add left or right (or both) hand for this dancer. In this case the terms 'left' and 'right' are meant according to the dancer's facing direction. A shape and size of the hands are computed so two dancers in a couple or mini-wave have joined hands, also four people in a star have inner hands nicely joined.

Using Hands with dancers facing h,v or x is not defined.

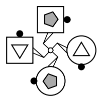
Example 1: Right-hand wave with joined hands drawn

```
A RH    n 0 0 b
A RH LH s 1 0 g
B RH LH n 2 0 b
B RH    s 3 0 g
```



Example 2: Right-hand star with inner hands joined

```
A RH n 0 0.5 b
A RH s 1 0.5 g
C RH e 0.5 1 b
C RH w 0.5 0 g
```



If you want to create more difficult diagrams you may use the fact that direction names 'n','s','e' and 'w' are the abbreviations of numbers only: n=90, w=180, s=270 and e=0. You can use another values to create dancers facing SE, SSE and so on :-) See also the 'rot' command below.

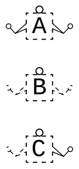
Note that 'h', 'v' and 'x' are more complicated macros – see Chapter Technical Notes.

2.4. Advanced phantoms

The next dancer parameter you can use is 'ph' for phantom. This is the first time when the order of parameters is important. 'ph' parameter changes all parts of dancer written on this line before 'ph' command to its phantom variation.

Example 1: See the difference

```
%%BoundingBox: 0 0 27 69
2 23 div  0 t        % enlarge and shift the diagram so hands will not touch
%                    % the diagram edge - don't worry about that

(A) ph LH RH n 0 2 b  % creates a phantom dancer with normal hands
(B) LH RH ph n 0 1 b  % creates a phantom dancer with phantom hands
(C) LH ph RH n 0 0 b  % creates a phantom dancer with phantom left hand and
%                     % normal right hand
```



2.5. Challenge commands

The folowing commands are standalone – placed on their own lines. They create new objects independent on any other dancer or change some general option. If they change some option, all following dancers are affected while the previous ones not – see the example near the 'rot' command.

2.5.1. bw

If you use graphical signs for dancers and don't have a printer with high quality of gray halftones printing, you may need a 'bw' command. It has no parameters, it simply switch the gray print to black&white one.

2.5.2. rot

This command is used to draw diagonal formations. Format of this standalone command (placed at its own line) is:

⟨*angle*⟩ ⟨*x*⟩ ⟨*y*⟩ *rot*

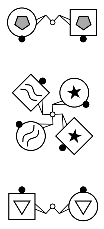
angle:
how many degrees (counted in the same way as for n,s,w and e directions) the rest of formation should be rotated

x,y:
coordinates of rotation center

Remember that sometimes you have to recompute BoundingBox parameters.

Example 1: Star in the middle rotated 1/8 (45 degrees)

```
C LH s 0 3 g
C RH s 1 3 b
A RH n 0 0 b
A LH n 1 0 g
45 0.5 1.5 rot
B RH n 0   1.5 g
D RH s 1   1.5 g
B RH e 0.5 2   b
D RH w 0.5 1   b
```



2.5.3. frame

Sometimes you need to emphasis some subformation. This can be done with 'frame' command:

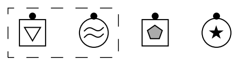
⟨*x1*⟩ ⟨*y1*⟩ ⟨*x2*⟩ ⟨*y2*⟩ *frame*

x1,y1:
coordinates of the dancer in the left bottom corner of the frame

x2,y2:
coordinates of the dancer in the right top corner of the frame

Example 1: Line with the left couple emphasized

```
A n 0 0 b
B n 1 0 g
C n 2 0 b
D n 3 0 g
0 0 1 0 frame
```



2.5.4. txt

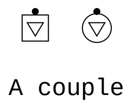
‘txt’ command is used to place a centered text label at some position:

⟨*(text)*⟩ ⟨*x*⟩ ⟨*y*⟩ *txt*

x, y: coordinates
Coordinates of the center of the text
(text): text of the label
Label text, enclosed in parenthesis

Example 1: Sdiagram with a label

```
A n 0 1 b
A n 1 1 g
(A couple) 0.5 0 txt
```



2.6. NOL: Hexagons

There is a support for hexagon diagrams also. Two standalone commands for facilitation of creating the hexagons are ‘hexinit’ and ‘nextthird’.

Syntax of ‘hexinit’ and ‘nextthird’ commands:

⟨*x*⟩ ⟨*y*⟩ *hexinit*

x, y: coordinates
Coordinates of the center of the hexagon formation

nextthird

Command ‘nextthird’ doesn’t have any parameter.

Firstly, the center of the formation is set with ‘hexinit’ command. Then draw one third of the dancers’ group (typicaly 4 dancers). Then use the ‘nextthird’ command to rotate the coordinate system around the defined center about 120 degrees (1/3), draw the second group of dancers, rotate using ‘nextthird’ again and draw the last part of dancers.

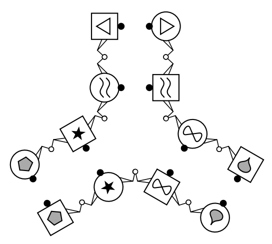
In case of symetrical diagram all three groups contain dancers with the same coordinates, only the dancers’ labels differs in each group.

Of course, you need to compute BoundingBox parameters as well as hexinit parameters. In the following examples, there are both BoundingBox parameters and user data shown, place it at according places in your eps file.

Example 1: Hexagon Facing Lines

Watch the perfectly connected hands and no need of special constants for coordinates!

```
%%BoundingBox: 0 0 105 92
1.8 1.3 hexinit

A RH e 0 2 b
A LH w 1 2 g
B LH RH e 0 1 g
```

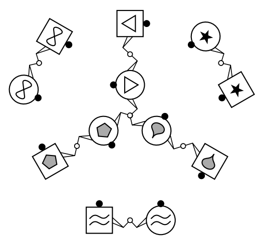


```
B LH RH w 1 1 b

nextthird

C RH e 0 2 b
C LH w 1 2 g
D LH RH e 0 1 g
D LH RH w 1 1 b

nextthird

E RH e 0 2 b
E LH w 1 2 g
F LH RH e 0 1 g
F LH RH w 1 1 b
```

Example 2: Hexagon Quater Tag

In case dancers stay on the center line you have to use a constant 0.707 instead of 0.5 for y coordinate. Watch the nicely connected hands again!

```
%%BoundingBox: 0 0 100 92
1.6 1.6 hexinit

A RH e 0.5 1.707 b
A LH RH w 0.5 0.707 g
B RH n 0 -1.5 b
B LH n 1 -1.5 g

nextthird

C RH e 0.5 1.707 b
C LH RH w 0.5 0.707 g
D RH n 0 -1.5 b
D LH n 1 -1.5 g

nextthird

E RH e 0.5 1.707 b
E LH RH w 0.5 0.707 g
F RH n 0 -1.5 b
F LH n 1 -1.5 g
```



3. Technical Notes

This chapter describes the PostScript background, so the PostScript knowledge is expected.

3.1. How this works

Each parameter (and “dancer name” which are PostScript macro command also) is defined as a macro which puts one {} block on the stack. Directly executable macros are used at the ends of lines: ‘b’, ‘g’ and standalone commands ‘bw’, ‘rot’, ‘txt’, ‘frame’, ‘hexinit’ and ‘nextthird’.

‘b’ and ‘g’ macros expect 4 parameters on the stack (from the top to bottom): y_coord, x_coord, angle of rotation and {} block drawing the dancer. Parameters of standalone commands are shown in their chapters.

Macros ‘n’, ‘s’, ‘e’ and ‘w’ expands to one number which means the angle to rotate from east orientation.

Macros ‘h’ and ‘v’ expands to two parts: a chain command (see below) which draws the opposite nose and an angle to rotate.

If something more should be exectuted, the fourth parameter of ‘b’ and ‘g’ (counted from the top of the stack) must be {} block which executes its own code and then get and execute the next parameter. This is called “chaining” because typical dancer’s line after the first PostScript expansion looks like this:

{ *...PS code drawing label...* } { *...param1 PS code... chain* } { *...param2 PS code... chain* } *...* ⟨*angle*⟩ ⟨*x*⟩ ⟨*y*⟩ *b*

PS code:
: raw PostScript code making the effect (draw dancer label, draw dancer’s hands, switch to phantom version...)

chain:
: PostScript macro ‘chain’ which does the effect of reading and executing the PostScript blocks in the reverse order than it is usual in PostScript language

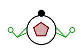
angle, x, y: usual part of SDIA dancer definition
: angle is one of e,w,n,s,x,v or h; x and y are dancer coordinates

Examples in this and following chapters show how the ‘clr’ sdia command was implemented.

The setrgbcolor PostScript command needs three numbers betveen 0 and 1, one for red, green and blue part of color. We know we should put it to ‘{ ... chain }’ scheme so for creating a dancer with dark red sign in it and with green hands we write the following.

Example 1: Colorized dancer – the first version

```
%%BoundingBox: 0 0 27 27
2 23 div  0 t          % enlarge and shift the diagram so hands will not touch
%                      % the diagram edge

C 0.5 0 0 { setrgbcolor chain } LH RH 0 0.6 0 { setrgbcolor chain } n 0 0 g
```



3.2. Making own macros

You can add a macro command for your own dancer parameter – it must put the block of PS code in “{ ... chain}” scheme on the stack.

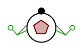
You can also create new graphical dancer label – “chain terminator”. These macros must put the block of PS code on the stack also but the block doesn’t contain the final chain command. It is a good idea to enclose the inner code between gsave..grestore pair – see sdia.eps code for useful macros and examples.

The coordinate system used in “label” macros is scaled by 0.004347 (=1/230). In this system, a size of the boy’s square is 49x49 and the diameter of the girl’s circle is 54.

Example 1: Colorized dancer – the second version

The previous example using PostScript macro for better readibility:

```
%%BoundingBox: 0 0 27 27
2 23 div  0 t         % enlarge and shift the diagram so hands will not touch
%                     % the diagram edge

/clr{{setrgbcolor chain}}!            %  ‘!’ is an alias to ‘bind def’

C 0.5 0 0 clr LH RH 0 0.6 0 clr n 0 0 g
```



Example 2: New dancer label “I” (simple x-type cross):

```
/I{{q newpath -30 -30 m 60 60 l -30 30 m 60 -60 l ss}}!
% q=gsave, m=moveto, l=rlineto, ss={th setlinewidth stroke grestore}

I n 0 0 g
```

4. Final word

Enjoy it! I hope you’ll create many nice diagrams for your articles, books and teaching material – or just for fun.

All comments and improvements are welcome. Feel free to send them to my e-mail address:

Milan Vancura ⟨milan@ucw.cz⟩