1. Installation

Given that all this stuff is still at development stage, the prefered way to install it is often to use the CVS version. This will allow you to get the latest innovations, and you may be able to help fixing the last bugs.

In order to get the sources from CVS, you just have to issue these two lines:

```
cvs -d:pserver:anonymous@cvs.enlightenment.sourceforge.net:/cvsroot/enlightenment login
cvs -z3 -d:pserver:anonymous@cvs.enlightenment.sourceforge.net:/cvsroot/enlightenment co e17
```

These two lines will download everything from the Enlightenment's repository. In the toplevel directory, you'll find two subdirectories: *libs*, which contains every e17-specific libraries, and *apps* which contains several programs and enlightenment.

Every library in *libs* can be compiled and installed with the following sequence:

```
./autogen.sh
make -11.656 Td[(cvs)-6su -c "make -600(install")]TJ/F27 9.963 Tf 0 -22.914 Td[(Ho)25(we)25(v)15(er)
```

- *RENDER_METHOD_3D_HARDWARE*, Evas will use the accelerated facilities of your 3D graphic card. This is certainly the fastest method if the graphic card does present an OpenGL acceleration, otherwise, it could be the slowest one! So use this method only is you do own such a card.
- *RENDER_METHOD_IMAGE*, with this method, Evas will only be able to draw into an Imlib2 picture, without actually displaying the result on the screen. This assumption will let Evas to do some optimisations. Nevertheless, this methood isn't very used in a regular application.

After that, in order to show the whole thing, we must link the canvas to a Drawable (i.e. a Window or a Pixmap). This

```
Window                window;
Evas                  canvas;
Evas_Object           object, line;
int                   i;

/* Canvas creation */
canvas = evas_new();
/* rendering method */
evas_set_output_method(canvas, RENDER_METHOD_ALPHA_SOFTWARE);
/* let's get three X-Window important variables */
display = XOpenDisplay(NULL);
visual = evas_get_optimal_visual(canvas, display);
colormap = evas_get_optimal_colormap(canvas, display);

/* Then, we create a suitable window */
att.colormap = colormap;
window = XCreateWindow(display,
                       RootWindow(display, DefaultScreen(display)),
                       0, 0, 500, 500, 0,
                       imlib_get_visual_depth(display, (canva1,
                       InputOutput, (ca, CWColormap, &att);
XMapWindow(display, window);
XSync(display, False);

/* link the canvas to the window */
evas_set_output(canvas, display, window, (ca, colormap);

/* The size of the canvas should be eq0cas to the window's size */
evas_set_output_size(canvas, 500, 500);

/* Let's create some transluscent rectangles */
for (i=25 ; i<250 ; i+=50)
{
  object = evas_add_rectangle(canvas);
  evas_show(canvas, object);
  evas_move(canvas, object, 250-i, i);
  evas_resize(canvas, object, i*2, (250-i)*2);
  evas_set_color(canvas, object, i, i, 255-i, 150);
}

/* a little animation, changing the canvas scale factor */
for (i=500 ; i<1000 ; i+=50)
{
  evas_set_output_viewport(canvas, 0, 0, i, i);

  evas_render(canvas);
}
evas_set_output_viewport(canvas, 0, 0, 500, 500);
evas_render(canvas);

/* At last, we move a line arosaT: */
line = evas_add_line(canvas);
evas_show(canvas, line);
```

```
evas_set_color(canvas, line, 255,255,255,255);
for (i=0 ; i<360 ; i+=5)
{
  evas_set_line_xy(canvas, line,
                   50+50*sin(i*2*M_PI/180.0),
                   50+50*cos(i*3*M_PI/180.0),
                   450+50*sin(i*4*M_PI/180.0),
                   450+50*cos(i*5*M_PI/180.0));
  evas_update_rect(canvas, 0, 0, 500, 500);
  evas_render(canvas);
}
resr511.(J0542g0ect(9)]TJ -xor
```