```
%scout

window liftvis;
point p1,p2;

p1 = {10, 10};
p2 = {310, 510};

liftvis = {
        box: [p1, p2],
        pict: "simulation",
        type: DONALD,
        sensitive: ON,
        border: 1,
        xmin: 20,
        xmax: 620,
        ymin: 0,
        ymax: 1000
};

screen = < liftvis >;

%donald
viewport dummyport

############
#LIFT USER#
############

openshape man
within man {

        circle head
        line body, leftarm, rightarm, leftleg, rightleg
        point manpos
        int rad

        manpos = {0,0}
        rad = 15
        head = circle(manpos, rad)
        body = [manpos - {0,rad}, manpos - {0,50}]
        leftarm = [manpos - {0,rad}, manpos - {20,40}]
        rightarm = [manpos - {0,rad}, manpos - {-20,40}]
        leftleg = [manpos - {0,50}, manpos - {20,70}]
        rightleg = [manpos - {0,50}, manpos - {-20,70}]

}


###########
#LIFT CAR#
###########
openshape box
within box {
        int width, length
        point p, q, b, d
        line top, bot, left, right

        b = {0, 0}
        d = b + {width, 0}
```

```
        p = b + {0, length}
        q = b + {width, length}

        width, length = 100, 100

        top = [p, q]
        bot = [b, d]
        left = [p, b]
        right = [q, d]

}

viewport simulation

int floor, liftfloor
floor = 1
liftfloor = 1

point carpos
carpos = {100, 50+(180*(liftfloor-1)) }

boolean car1,car2,car3,car4,car5
car1=false
car2=false
car3=false
car4=false
car5=false

openshape car
within car {

        point corner
        corner = ~/carpos
        shape car
        int X, Y
        real ratio
        ratio = 0.35
        car = trans( scale(~/box, 5*ratio), corner.1, corner.2)
        X = 175         # = 500 * ratio (but type problems in defn)

        openshape button1
        within button1 {
                shape button1
                boolean light
                light = ~/~/car1
                button1 = trans( scale(~/~/box, ~/ratio),
~/corner.1+~/X,~/corner.2)
        }

        openshape button2
        within button2 {
                shape button2
                button2 = trans( scale(~/~/box, ~/ratio), ~/corner.1+~/X,
~/corner.2+~/ratio*100)
                boolean light
                light = ~/~/car2
        }

        openshape button3
        within button3 {
                shape button3
```

```
                button3 = trans( scale(~/~/box, ~/ratio), ~/corner.1+~/X,
~/corner.2+~/ratio*200)
                boolean light
                light = ~/~/car3
        }

        openshape button4
        within button4 {
                shape button4
                button4 = trans( scale(~/~/box, ~/ratio), ~/corner.1+~/X,
~/corner.2+~/ratio*300)
                boolean light
                light = ~/~/car4
        }

        openshape button5
        within button5 {
                shape button5
                button5 = trans( scale(~/~/box, ~/ratio), ~/corner.1+~/X,
~/corner.2+~/ratio*400)
                boolean light
                light = ~/~/car5
        }


}
? A_car_button1_button1 is (_car_button1_light) ? "linewidth=5" :
"linewidth=0";
? A_car_button2_button2 is (_car_button2_light) ? "linewidth=5" :
"linewidth=0";
? A_car_button3_button3 is (_car_button3_light) ? "linewidth=5" :
"linewidth=0";
? A_car_button4_button4 is (_car_button4_light) ? "linewidth=5" :
"linewidth=0";
? A_car_button5_button5 is (_car_button5_light) ? "linewidth=5" :
"linewidth=0";

##########
#LANDINGS#
##########

int ceiling, wall
ceiling = 950
wall = 340
real ratio
ratio = 1.8

boolean button1,button2,button3,button4,button5
button1 = false
button2 = false
button3 = false
button4 = false
button5 = false

openshape floor1
within floor1 {
        shape floor1
        floor1 = trans( scale(~/box, ~/ratio), ~/wall, ~/ceiling-
~/ratio*500)
        openshape button
        within button {
```

```
                shape button
                boolean light
                light = ~/~/button1
                button = trans( scale(~/~/box, 0.2), ~/~/wall, ~/~/ceiling-
~/~/ratio*450)
        }
}

openshape floor2
within floor2 {
        shape floor2
        floor2 = trans( scale(~/box, ~/ratio), ~/wall, ~/ceiling-
~/ratio*400)
        openshape button
        within button {
                shape button
                boolean light
                light = ~/~/button2
                button = trans( scale(~/~/box, 0.2), ~/~/wall, ~/~/ceiling-
~/~/ratio*350)
        }
}

openshape floor3
within floor3 {
        shape floor3
        floor3 = trans( scale(~/box, ~/ratio), ~/wall, ~/ceiling-
~/ratio*300)
        openshape button
        within button {
                shape button
                boolean light
                light = ~/~/button3
                button = trans( scale(~/~/box, 0.2), ~/~/wall, ~/~/ceiling-
~/~/ratio*250)
        }
}

openshape floor4
within floor4 {
        shape floor4
        floor4 = trans( scale(~/box, ~/ratio), ~/wall, ~/ceiling-
~/ratio*200)
        openshape button
        within button {
                shape button
                boolean light
                light = ~/~/button4
                button = trans( scale(~/~/box, 0.2), ~/~/wall, ~/~/ceiling-
~/~/ratio*150)
        }
}

openshape floor5
within floor5 {
        shape floor5
        floor5 = trans( scale(~/box, ~/ratio), ~/wall, ~/ceiling-
~/ratio*100)
        openshape button
        within button {
                shape button
```

```
                boolean light
                light = ~/~/button5
                button = trans( scale(~/~/box, 0.2), ~/~/wall, ~/~/ceiling-
~/~/ratio*50)
        }
}

? A_floor1_button_button is (_floor1_button_light) ? "linewidth=5" :
"linewidth=0";
? A_floor2_button_button is (_floor2_button_light) ? "linewidth=5" :
"linewidth=0";
? A_floor3_button_button is (_floor3_button_light) ? "linewidth=5" :
"linewidth=0";
? A_floor4_button_button is (_floor4_button_light) ? "linewidth=5" :
"linewidth=0";
? A_floor5_button_button is (_floor5_button_light) ? "linewidth=5" :
"linewidth=0";


%donald
line door1, door2, door3, door4, door5
door1 = [ {wall - 15, ceiling - ratio*500}, {wall - 15, ceiling -
ratio*405}]
door2 = [ {wall - 15, ceiling - ratio*400}, {wall - 15, ceiling -
ratio*305}]
door3 = [ {wall - 15, ceiling - ratio*300}, {wall - 15, ceiling -
ratio*205}]
door4 = [ {wall - 15, ceiling - ratio*200}, {wall - 15, ceiling -
ratio*105}]
door5 = [ {wall - 15, ceiling - ratio*100}, {wall - 15, ceiling - ratio*5}]

%eden
A_door1 is "color=red,linewidth=3";
A_door2 is "color=red,linewidth=3";
A_door3 is "color=red,linewidth=3";
A_door4 is "color=red,linewidth=3";
A_door5 is "color=red,linewidth=3";
%eden
A_door2 is "color=grey,linewidth=3";

%donald

label whoat0, whoat1, whoat2, whoat3, whoat4, whoat5

# model the users of the lift - X, Y, Z and I
# have a boolean inliftX to record whether X is in lift - if not, then
# will appear where specified by floorX, which takes values 1-5 and 7
(nowhere)
# inliftX is defined to be the case when floorX has value 0

point one, put
one = {wall + 100, 10}

shape personX
boolean inliftX
int floorX
floorX = 7
inliftX = floorX == 0
put = {carpos.1 + 100, carpos.2 + 140}
personX = if inliftX then trans(scale(man, 2), put.1, put.2) else
trans(scale(man, 2), one.1, one.2 + (floorX * 180))
```

```
shape personY
boolean inliftY
int floorY
floorY = 7
inliftY = floorY == 0
put = {carpos.1 + 100, carpos.2 + 140}
personY = if inliftY then trans(scale(man, 2), put.1, put.2) else
trans(scale(man, 2), one.1, one.2 + (floorY * 180))

shape personI
boolean inliftI
int floorI
floorI = 7
inliftI = floorI == 0
put = {carpos.1 + 100, carpos.2 + 140}
personI = if inliftI then trans(scale(man, 2), put.1, put.2) else
trans(scale(man, 2), one.1, one.2 + (floorI * 180))

shape personZ
boolean inliftZ
int floorZ
floorZ = 7
inliftZ = floorZ == 0
put = {carpos.1 + 100, carpos.2 + 140}
personZ = if inliftZ then trans(scale(man, 2), put.1, put.2) else
trans(scale(man, 2), one.1, one.2 + (floorZ * 180))

%eden
installeddi();

%eddi
users(id char, loc int, dest int);
users << ["X", 1, 3], ["Y",1, 4], ["I", 1, 5], ["Z", 3, 1];

%eden

users is [[["string","int","int"],"id","loc","dest"],["X",locX,
destX],["Y",locY,destY],["I",locI,destI],["Z",locZ,destZ]];

locX = locY = locI = 7;
destX = destI = 4;
destY = 5;
locZ = 7;
destZ = 1;

_floorX is locX;
_floorY is locY;
_floorI is locI;
_floorZ is locZ;

at0 is project(select_wrapper(users, "loc==0"), ["id"]);     /* in lift */
at1 is project(select_wrapper(users, "loc==1"), ["id"]);
at2 is project(select_wrapper(users, "loc==2"), ["id"]);
at3 is project(select_wrapper(users, "loc==3"), ["id"]);
at4 is project(select_wrapper(users, "loc==4"), ["id"]);
at5 is project(select_wrapper(users, "loc==5"), ["id"]);
at7 is project(select_wrapper(users, "loc==7"), ["id"]);     /* nowhere */

func findlab {
        para idsatloc;
```

```
        auto i, result;
        result = "";
        for (i=2; i<=idsatloc#; i++)
                result = result // idsatloc[i][1];
        return result;
}

_whoat0 is label(findlab(at0), cart(dot1(_carpos) + 100, dot2(_carpos) +
140));
_whoat1 is label(findlab(at1), cart(_wall + 100, 10 + (1 * 180)));
_whoat2 is label(findlab(at2), cart(_wall + 100, 10 + (2 * 180)));
_whoat3 is label(findlab(at3), cart(_wall + 100, 10 + (3 * 180)));
_whoat4 is label(findlab(at4), cart(_wall + 100, 10 + (4 * 180)));
_whoat5 is label(findlab(at5), cart(_wall + 100, 10 + (5 * 180)));

%donald
boolean open1, open2, open3, open4, open5
open1, open2, open3, open4, open5 = false, false, false, false, false

%eden
coldoor1 is (_open1)?"grey":"red";
A_door1 is "color=" // coldoor1 // ",linewidth=3";
coldoor2 is (_open2)?"grey":"red";
A_door2 is "color=" // coldoor2 // ",linewidth=3";
coldoor3 is (_open3)?"grey":"red";
A_door3 is "color=" // coldoor3 // ",linewidth=3";
coldoor4 is (_open4)?"grey":"red";
A_door4 is "color=" // coldoor4 // ",linewidth=3";
coldoor5 is (_open5)?"grey":"red";
A_door5 is "color=" // coldoor5 // ",linewidth=3";

_liftfloor = _liftfloor;
/* this makes _liftfloor a read/write variable, supporting _liftfloor++;
etc */


/* some sample definitions to affect the state of the lift */

_button1 = 1;
locX = 1;
locY = 4;
locZ = 0;
_open4 = 1;
_liftfloor = 4;
_car5 = 1;
```