Drag and Drop Icons and Their GML Equivalents for Version 6.1

Coding, Sprites, Logo © Mark Overmars, http://www.gamemaker.nl/
Prepared by D. Eugene Perry, http://www.blackratstudios.com/

Table of Contents

Notes:

Reference Colours

These are the colours that are used on the following pages and what they mean…

Green - Comments
Purple – User input
Blue – Built in Variables or statements.

Assigning statements to other objects

Drag and drop icons allow you to assign statements to other objects. You can do this in code with the following...

```
//to assign to another object...
with (object) {
//actions here
}
```

```
//To assign to the other object in a collision...
with (other) {
//actions here
}
```

```
//To assign the statement to that object...
with (self) {
//actions here
}
```

Using 'ID' as opposed to 'id'

Please note that throughout the document I show examples of assigning code to other objects from a separate object. In order to make it easier I use the capitalization, 'ID' which Game Maker recognizes as a local variable. This is not to be confused with 'id' which is an inherited variable that will give you the current object's unique id number.

Move Tab

_____

Move section

⊛'Start moving in a direction'

⊛'Set the direction and speed of motion'
motion_set(direction,speed);

↗'Move in the direction of a point'
move_towards_point(x,y,speed);

⮞'Set the horizontal speed'
hspeed=speed;

⬇'Set the vertical speed'
vspeed=speed;

⬇'Set the gravity'
gravity_direction=direction;
gravity=amount;

↩'Reverse horizontal direction'
hspeed=-hspeed; //actual code.

↪'Reverse vertical direction'
vspeed=-vspeed; //actual code.

⇨'Set the friction'
friction=amount;

Jump Section

↷'Jump to a given position'
x=value;
y=value;

'Jump to the start position'
x=xstart; //actual code.
y=ystart; //actual code.

'Jump to a random position'
move_random(1,1); //actual code. The 1 and 1 in the code are the hsnap and vsnap positions.

'Snap to a grid'
move_snap(hsnap,vsnap);

'Wrap when moving outside'
move_wrap(hort,vert,margin);
// This code should be placed in the outside room event. Set hort (horizontally) and vert (vertically) to either 1 for true or 0 for false. Set margin to how far outside the room the instance must be before the action happens.

'Move to a contact position'
move_contact_solid(dir,maxdist) //for solid objects
move_contact_all(dir,maxdist) //for non solid objects //dir=direction, maxdist=maximum distance.

'Bounce against objects'
move_bounce_solid(advanced); //for solid objects, advanced=advance bounce(0 or 1).
move_bounce_all(advanced); //for all objects, advanced=advance bounce(0 or 1).

Paths Section

'Set path for the instance'
path_start(path,speed,endaction,absolute);

'End the path for an instance'
path_end();

'Set the position on the path'
path_position=value; //Must lie between 0 and 1.

'Set the speed for the path'
path_speed=value; //pixels per step.

Steps Section

'Perform a step towards a point'
mp_linear_step(x,y,stepsize,checkall);
// stepsize is in pixels. Checkall can be either 1 for stopping when hitting any object, or 0 for only solid objects.

'Step towards a point avoiding objects'
mp_potential_step(x,y,stepsize,checkall);

Main1 Tab

_____

Objects Section

'Create an instance of an object');
instance_create(x,y,object0); //use x and y variables for relative.

'Create an instance of an object with a motion
//No equivalent, but you can use the following code.
ID = instance_create(x,y,object1);
with (ID) motion_set(direction,speed);

'Create instance of random object'
item[0]=object0;
item[1]=object1;
item[2]=object2;
item[3]=object3;
instance_create(x,y,item[floor(random(4))]);
//Assign each object name to an array, then use the random statement to choose.

'Change the instance'
instance_change(obj,perf);
//perf(1 or 0)is whether or not to perform create and destroy events.

'Destroy the instance'
instance_destroy();

'Destroy instance at position'
position_destroy(x,y);

Sprite Section

'Change the sprite'
sprite_index=sprite0;

'Transform the sprite'
image_xscale=value; //horizontal scaling of sprite.
image_yscale=value; //verticle scaling if the sprite.
image_angle=value; //angle the sprite.
image_xscale=-1; //flip the sprite horizontally, actual code.
image_yscale=-1; //flip the sprite vertically, actual code.

'Set sprite blending'
image_blend=colour;
image_alpha=value; //from 0 to 1, 1 being opaque.

Sounds Section

'Play a sound'
sound_play(sound); //plays sound once.
sound_loop(sound); //loops sound.

'Stop a sound'
sound_stop(index); //Stops the indicates sound. If there are multiple sounds with this index playing simultaneously, all will be stopped.

'If a sound is playing'
if sound_isplaying(sound)=true {
// actions here.
}

Rooms Section

//If you wish to use transitions with the following statements call this statement first…

transition kind=value;

Values for above as follows...

0 = no effect
1 = Create from left
2 = Create from right
3 = Create from top
4 = Create from bottom
5 = Create from center
6 = Shift from left
7 = Shift from right
8 = Shift from top
9 = Shift from bottom
10 = Interlaced from left
11 = Interlaced from right
12 = Interlaced from top
13 = Interlaced from bottom

'Go to previous room'
room_goto_previous();

'Go to next room'
room_goto_next();

'Restart the current room'
room_restart();

'Go to a different room'
room_goto(room);

'if previous room exists'
if room_previous(room)<>-1 then {
// actions here
}
//'room' is constant variable for current room. Actual code.

'If next room exists'
```
if room_next(room)<>-1 then {
// actions here.
}
//'room' is constant variable for current room. Actual code.
```

Main 2 Tab

Timing Section

'Set an alarm'
```
alarm[0]=value; //set 0 from 0 to 7 for alarm.
```

'Sleep for a while'
```
sleep(numb); //'numb' is in milliseconds.
```

'Set a timeline'
```
timeline_index=timeline;
```

'Set timeline position'
```
timeline_position=value;
```

Info Section

'Display a message'
```
show_message('Hello');
```

'Show the game info'
```
show_info();
```

'Show a video'
```
show_video(fname,full,loop); //full and loop are either 1 or 0 for yes or no.
```

Game Section

'Restart the game'
game_restart();

'End the game'
game_end();

'Save the game'
game_save(fname);
//fname is the name of the save file. Place it in quotes.

'Load the game'
game_load(fname);
//fname is the name of the save file to load. Place it in quotes.

Resources Section

'Replace a sprite from a file'
sprite_replace(ind,fname,imgnumb,precise,transparent,smooth,preload,xorig,yorig);
// precise, transparent,smooth,preload are all 1 or 0 for yes or no.

'Replace a sound from a file'
sound_replace(index,fname,kind,loadonuse);
//loadonuse is 1 or 0 for yes or no.
//Kind is one of the following...
0-normal
1-background
2-3d
3-mmplayer

'Replace a background from a file'
background_replace(ind,fname,transparent,smooth,preload);
// transparent,smooth and preload are all 1 or 0 for yes or no.

Control tab

_____

Questions Section

'If a position is collision free'
```
if place_free(x,y) {//for solid
//actions here.
}
if !place_empty(x,y) //for all
//actions here.
}
```

'if there is a collision at position'
```
if collision_point(x,y,obj,prec,notme) {
//actions here.
if collision_rectangle(x1,y1,x2,y2,obj,prec,notme) {
//actions here.
}
if collision_circle(xc,yc,radius,obj,prec,notme) {
//actions here.
}
if collision_ellipse(x1,y1,x2,y2,obj,prec,notme) {
//actions here.
}
if collision_line(x1,y1,x2,y2,obj,prec,notme) {
//actions here.
}
```

'If there is an object at position'
```
if position_meeting(x,y,obj) {
//actions here.
}
```

'If the number of instances is a value {
```
if instance_number(obj)=value {
//actions here.
}
```

'With a chance perform the next action
```
if floor(random(value))=0 {
//actions here.
}
```

'If the user answers yes to a question'
```
if show_question('Do you want to do this?') {
// actions here.
}
```

'If an expression is true'
```
if (the expression) {
// actions here.
}
```

'If a mouse button is pressed'
```
if mouse_check_button(numb) {
//actions here.
}
// numb can be mb_none,mb_left, mb_middle,mb_right.
```

'If an instance is aligned with a grid'
```
if place_snapped(value,value) {
//actions here.
}
```

Other Section

'Start of block'

'End of block'

'Else'
```
//All above are part of if, else statements example...
if x=50 {
hspeed=2;
vspeed=-2;
}
else {
motion_set(90,1);
}
```

'Exit this event'
exit;

'Repeat next action'
repeat (value) <statement>;
//example: repeat (10) instance_create(x,y,object0);

'Call the inherited event'
event_inherited();

Code Section

'Execute a piece of code'
//This is the icon that all coding is placed in.

'Call a script'
script_execute(ind,arguments);
// or call a script in code by the script name and the arguments in () beside it…

 'Put some comment'
//Enter a comment in code by putting '//' followed by the comment.

Variables Section

'Set a variable'
// Set a variable by either using a built-in variable or by using your own.
example...
health=50;
lives=3;
name='Gordon';
//Use 'global.' for your own variables that are to be used by more than one object...
global.name='Gordon';
//You do not need to use global for built-in variables like 'lives', or 'score'.

'If a variable has a value'
//Use an if statement to check this. Example...
if lives=0 {
//actions here
}

VAR 'Draw the value of a variable'
draw_text(x,y,global.name);
draw_text(x,y,lives);

Score Tab

_____

Score Section

'Set the score'
score=value;

'If score has a value'
if score=value {
//actions here.
}

'Draw the value of the score'
draw_text(x,y,'Score: ' + string(score));

'Show the highscore table'
highscore_set_background(back); //set with background image.
highscore_set_border(show); //1 or 0 for yes or no.
highscore_set_colors(back,new,other); //set colours for background,new entry,
other entries.
highscore_set_font(name,size,style); //set style to 0=normal, 1=bold, 2=italic,
3=bold italic.
highscore_show(numb); //This actually shows the table, with numb being the new
score to add if it is high enough.
//Note: there are many other controls for the high score table. These are just the
ones used in the drag and drop.

'Clear the highscore table
highscore_clear();

Lives Section

'Set the number of lives;
lives=value;

'If lives has a value'
if lives=value {
//actions here.
}
'Draw the number of lives'
draw_text(x,y,'Lives: ' + string(lives));

'Draw the lives as an image'
// no equivalent but you can use the following code in the draw event. sprite0 is
the sprite image. Set 'a' in the 5th line to however far apart you wish the images
to be on the screen in pixels...
var a;
a=0;
repeat(lives){
draw_sprite(sprite0,0,view_xview+a,view_yview);
a+=16;
}

Health Section

'Set the health'
health=value;

'If health is a value'
if health=value {
//actions here.
}

'Draw the health bar'
draw_healthbar(x1,y1,x2,y2,amount,backcol,mincol,maxcol,direction,showback,s
howborder);

🔲'Set the window caption info'
show_score=value; //set to 1 for yes, 0 for no.
caption_score=string;
show_lives=value; //set to 1 for yes, to 0 for no.
caption_lives=string;
show_health=value; //set to 1 for yes, to 0 for no.
caption_health=string;

Extra Tab

_____

Particles Section

🔲'Create the particle system'
index=part_system_create(); //Assign to an index (variable). Must be used in other functions.

🔲'Particle system destroy'
part_system_destroy(index);

🔲'Clear all particles from the system'
part_system_clear(index);

🔲'Create a particle'
index=part_type_create(); //Assign to an index.
part_type_shape(index,shape); //see manual for shape types.
part_type_size(index,size_min,size_max,size_incr,size_rand);
part_type_color(index,color_start,color_middle,color_end);
//there are other functions forparticles; these just cover the Drag and Drop.

🔲'Set the color for a particle type'
part_type_color1(ind,color1)
//Indicates a single color to be used for the particle.
part_type_color2(ind,color1,color2)
//Specifies two colors between which the color is interpolated.
part_type_color3(ind,color1,color2,color3)
//Similar but this time the color is interpolated between three colors that represent the color at the start, half-way, and at the end.
part_type_color_mix(ind,color1,color2)
//With this function you indicate that the particle should get a color that is a random mixture of the two indicated colors. This color will remain fixed over the lifetime of the particle.
part_type_color_rgb(ind,rmin,rmax,gmin,gmax,bmin,bmax)

//Can be used to indicate that each particle must have a fixed color but chosen from a range. You specify a range in the red, green, and blue component of the color (each between 0 and 255).
part_type_color_hsv(ind,hmin,hmax,smin,smax,vmin,vmax)
//Can be used to indicate that each particle must have a fixed color but chosen from a range. You specify a range in the hue saturation and value component of the color (each between 0 and 255).

'Set the life time for the particle'
part_type_life(index,life_min,life_max);

'Set the motion for the particle'
part_type_speed(index,speed_min,speed_max,speed_incr,speed_rand);
part_type_direction(index,dir_min,dir_max,dir_incr,dir_rand);

'Set the gravity for the particle'
part_type_gravity(index,grav_amount,grav_dir);

'Create secondary particles'
part_type_death(index,death_number,death_type);

'Create a particle emitter'
index=part_emitter_create(ps); //ps is the index of the particle system. You must assign this to an index.
part_emitter_region(ps,index,xmin,xmax,ymin,ymax,shape,distribution);
//ps is the index of the particle system. Index is the index of the emitter.

'Destroy an emitter'
part_emitter_destroy_all(ps) //ps is the index of the emitter.

'Burst a number of particles from an emitter'
part_emitter_burst(ps,index,parttype,number) ; //ps is the index of the particle system. Index is the index of the emitter. Parttype is the index of the particle.

'Stream particles from an emitter'
part_emitter_stream(ps,index,parttype,number);

CD Section

//You must call the function cd_init(); before calling other Cd functions.

'Play a CD'
cd_play(first,last);

'Stop the CD'
cd_stop();

'Pause the CD'
cd_pause();

'Resume the CD'
cd_resume();

'If a CD exists in the drive'
if cd_present()=true {
//actions here.
}

'If the CD is playing'
if cd_playing()=true {
//actions here.
}

Other Section

'Set the mouse cursor';
window_set_cursor(curs);
//this will set the cursor to a default setting (see manual for types)
to have a custom sprite as a cursor use the following statement…
cursor_sprite=sprite0;
//Change sprite0 to sprite name This does not appear in the help manual, but it is
valid for GM6. When using it though, set the actual cursor not to show else you
will have two cursors.

'Open a web page in a browser'
execute_shell('http//www.somepage.com',0);

Drawing tab

_____

Drawing Section
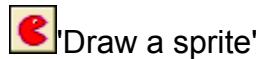
'Draw a sprite'
draw_sprite(sprite,subimage,x,y);

'Draw a background image'
draw_background(back,x,y) //single image.
draw_background_tiled(back,x,y); //tiled image.

'Draw a text'
draw_text(x,y,string);

'Draw text transformed'
draw_text_transformed(x,y,string,xscale,yscale,angle);

'Draw a rectangle'
draw_rectangle(x1,y1,x2,y2,outline); //Outline is 1 or 0 for yes or no.

'Draw a horizontal gradient'
draw_rectangle_color(x1,y1,x2,y2,col1,col2,col3,col4,outline);
//Set col1 and col4 to the left color. Set col2 and col3 to the second color.

'Draw a vertical gradient'
draw_rectangle_color(x1,y1,x2,y2,col1,col2,col3,col4,outline);
//Set col1 and col2 to the top color. Set col3 and col4 to the bottom color.

'Draw an ellipse'
draw_ellipse(x1,y1,x2,y2,outline); //outline is 1 or 0 for yes or no.

'Draw a gradient ellipse'
draw_ellipse_color(x1,y1,x2,y2,col1,col2,outline);
//col1 is the color in the middle. col2 is the color at the boundary.

'Draw a line'
draw_line(x1,y1,x2,y2);

'Draw an arrow'
draw_arrow(x1,y1,x2,y2,size); //size is in pixels.

Settings Section

'Set the color'
draw_set_color(col);
//See the manual for colors.
'Set the font'
draw_set_font(font);
draw_set_halign(halign); //Can be set to fa_left, fa_center, fa_right.
draw_set_valign(valign); //Can be set to fa_top, fa_middle, fa_bottom.

'Change full screen mode'
window_set_fullscreen(full); //Set to 0 for window, to 1 for fullscreen.

Other Section

'Take a snapshot of the image'
screen_save(filename);

'Create an effect'
effect_create_below(kind,x,y,size,color) //Creates an effect of the given kind (see above) at the indicated position. size give the size as follows: 0 = small, 1 = medium, 2 = large. color indicates the color to be used. The effect is created below the instances, that is, at a depth of 100000.
effect_create_above(kind,x,y,size,color) //Similar to the previous function but this time the effect is created on top of the instances, that is, at a depth of -100000.

//Following are the effect kinds to use in the above statements...

ef_explosion
ef_ring
ef_ellipse
ef_firework
ef_smoke
ef_smokeup
ef_star
ef_spark
ef_flare

ef_cloud
ef_rain
ef_snow

Thanks to the Following

Mark Overmars for making Game Maker (of course)
http://www.gamemaker.nl/

New Game Studios Representative
For the original suggestion that lead to this.

And many thanks to everyone who PM'd fixing my many errors. I wish I had remembered to jot down all your names but I rarely think ahead like that (shoot). But you know who you are so….thanks!