Printing

This extension package provides printing functionality for *Game Maker* games. A large collection of functions is provided to print text, shapes, images, and screen shots on the selected printer.

To use this package in a game, select the item **Select Extension Packages** in the **Resources** menu. In the form that appears click on the **GM Printing** item in the right list and press the button to move it to the
left list. That is all.

Below you find a list of all functions and constants that are available in this extension package. Note that all functions start with the prefix pr_.

Global Idea

The global idea behind printing is as follows. You start the printing job by calling the functions pr_begin_document(). After this there are functions to draw different shapes and texts at locations on the page. When you created the first page you call the function pr_newpage() after which you can do the printing on the next page. When you finished the final page you call the function pr_end_document(). The actual printing only starts at that moment. Here is a simple example:

```
pr_begin_document();
pr_set_font_size(32);
pr_draw_text(500,500,'Page 1');
pr_newpage();
pr_draw_text(500,500.'Page 2');
```

```
pr_end_document();
```

Please realize that the resolution of the printer is normally very high and is printer dependent. You have to take this into account when deciding which sizes to use.

Printing happens on the selected printer. If you want to let the user select a different printer you can use the print dialog functions that are described at the end of this document.

Handling Pages

Printing happens on pages. Before printing the first page the document must be begun and after describing the final page the document must be ended. The following functions exist for this:

pr_begin_document() This function start the description of the
document. It must be called before doing any printing.
pr_end_document() This function ends the description of the
document. It must be called before after the description. Only when
this function is called the actual printing happens.
pr_newpage() This function starts a new page.

pr_abort() This function aborts the printing document you are currently describing without printing it.

Printer Properties

To print things at the correct position on the page it is important to get information about the size of the page (in pixels). With the following functions you can obtain information about the printer and set certain properties:

pr_get_width() This function returns the width of a page in pixels.
pr_get_height() This function returns the height of the page in pixels.

pr_get_pixelsperinch() This function returns the number of pixels in one inch. In combination with the above functions you can determine the page size.

pr_get_orientation() This function returns the orientation of the page. It can either be pr_or_portrait or pr_or_landscape.

pr_set_orientation(orientation) This function sets the orientation
of the page. It can either be set to pr_or_portrait or to
pr_or_landscape.

pr_get_title() This function returns the title of the document as it is used in the printer queue.

pr_set_title(title) With this function you can set the title of the document as it is used in the printer queue.

pr_get_pagenumber() This function returns the number of the current page being described.

pr_get_printing() This function returns whether you are currently
describing a print document, that is pr_begin_document has been
called but not yet pr_end_document.

Pen, Brush, and Font

The way shapes are drawn depend on the pen, brush, and font. The pen determines how the outline of shapes are drawn. The brush determines how they are filled. And the font determines the font that is used when drawing text. Their properties can be set and retrieved with various functions. For the pen the following functions exists:

pr_set_pen_width(width) This function sets the with of the pen in pixels.

pr_set_pen_color(col) This function sets the color of the pen.
pr_set_pen_style() This function sets the style of the pen. You can
use the following values:

```
pr_ps_solid = the normal solid pen pr_ps_dash = a dashed line
pr_ps_dot = a dotted line pr_ps_dashdot = dash dot dash dot,
etc. pr_ps_dashdotdot = dash dot dot dash dot dot, etc.
pr_ps_clear = no line is drawn at all
```

Note that the way these styles look are printer dependent. So you are recommended not to use them, except for solid and clear. pr_get_pen_width() This function returns the width of the pen in pixels.

```
pr_get_pen_color() This function returns the color of the pen.
pr_get_pen_style() This function returns the style of the pen.
```

For the brush the following functions exist:

pr_set_brush_color(col) This function sets the color of the brush.
pr_set_brush_style() This function sets the style of the brush. You
can use the following values:

```
pr_bs_solid = the normal solid brush pr_bs_horizontal =
horizontal lines pr_bs_vertical = vertical lines pr_bs_cross =
horizontal and vertical lines pr_bs_updiagonal = diagonal lines
going up pr_bs_downdiagonal = diagonal lines going down
```

pr_bs_diagcross = diagonal lines crossing pr_bs_clear = no fill
is applied

Note that the way these styles look are printer dependent. So you are recommended not to use them, except for solid and clear. pr_get_brush_color() This function returns the color of the brush. pr_get_brush_style() This function returns the style of the brush.

For the font the following functions exist:

pr_set_font_name(name) This function sets the name of the font.
Better make sure the font exists.

pr_set_font_color(col) This function sets the color for the font.
pr_set_font_size(size) This function sets the size for the font. The font size is set in points, not in pixels. One inch corresponds to 72 points. So if you use font sizes the text will have exactly the same size independent of the type of printer.

pr_set_font_height(height) This function sets the height for the font
in pixels. You are recommended to set the size instead of the height.
pr_set_font_style(style) With this function the font style can be
set. The following styles exist:

```
pr_fs_normal = normal style pr_fs_bold = bold style
pr_fs_italic = italic style pr_fs_underline = underline style
pr_fs_strikeout = strikeout style
```

If you want a combination of these style, simply add up the values. pr_set_font_align(align) This function sets the font alignment. You can use the following values:

```
pr_fa_left = left aligned pr_fa_center = centered pr_fa_right
= right aligned
```

pr_set_font_angle(angle) This function sets the font angle in
degrees, counter-clockwise. (Not all fonts support this.)
pr_get_font_name() This function returns the font name.
pr_get_font_color() This function returns the font color.
pr_get_font_size() This function returns the font size.

```
pr_get_font_height() This function returns the font height.
pr_get_font_style() This function returns the current font style.
pr_get_font_align() This function returns the current font alignment.
pr_get_font_angle() This function returns the current font angle.
```

Printing Shapes and Text

This section described the various functions that are available for drawing shapes and text on the print document. The following functions are available for printing shapes:

pr_print_line(x0,y0,x1,y1) This function prints a line from position (x0,y0) to position (x1,y1) using the current pen settings. pr_print_arrow(x0,y0,x1,y1,size) This function prints an from position (x0,y0) to position (x1,y1) using the current pen and brush settings. size indicates the length of the arrow along the line. pr_print_rectangle(x0,y0,x1,y1) This function prints a rectangle with corners (x0,y0) and (x1,y1) using the current pen and brush settings.

 $pr_print_fillrect(x0,y0,x1,y1)$ This function prints a rectangle without an outline with corners (x0,y0) and (x1,y1) using the current brush settings.

pr_print_framerect(x0,y0,x1,y1) This function prints an outline rectangle with corners (x0,y0) and (x1,y1) using the current brush settings. (Note that the brush is used here, not the pen!) $pr_print_roundrect(x0,y0,x1,y1,x2,y2)$ This function prints a rounded rectangle with corners (x0,y0) and (x1,y1) using the current pen and brush settings. x2 and y2 are the size of the rounded regions in the horizontal and vertical direction.

pr_print_triangle(x0, y0, x1, y1, x2, y2) This function prints a triangle
with the indicated vertices.

 $pr_print_ellipse(x0,y0,x1,y1)$ This function prints an ellipse inside the rectangle with corners (x0,y0) and (x1,y1) using the current pen and brush settings.

 $pr_print_circle(x0,y0,r)$ This function prints a circle with center (x0,y0) and radius r using the current pen and brush settings.

 $pr_print_arc(x0,y0,x1,y1,x2,y2,x3,y3)$ This function draws an arc of an ellipse. (x0,y0) and (x1,y1) indicate the size of the ellipse. The arc is drawn following the perimeter of the ellipse, counterclockwise, from the starting point to the ending point. The starting point is the

intersection of the ellipse and the line from the center to (x2,y2). Then ending point is the intersection of the ellipse and the line form the center to (x3,y3).

pr_print_chord(x0,y0,x1,y1,x2,y2,x3,y3) This function draws a chord of an ellipse. This is the same as the arc defined with the previous function but this time it is closed with a straight line between the starting and ending point.

pr_print_pie(x0,y0,x1,y1,x2,y2,x3,y3) This function draws a pie of an ellipse. This is the same as the arc defined with the previous function but this time it is closed with two lines between the center and the starting and ending point.

pr_print_polygon_start() Start the definition of a polygon for printing.

pr_print_polygon_vertex(x0,y0) Adds a vertex to the polygon.
pr_print_polygon_end() Ends the definition of the polygon and
actually prints it.

The following functions are available for printing text:

pr_print_text(x,y,str) This function prints the string str at location
(x,y) using the current font settings. Note that the brush setting is
used to determine the background of the font. If you want to make
the background transparent, set the brush style to pr_bs_clear.
pr_text_width(str) This function returns the width of the string str
when printed using the current font settings.

pr_text_height(str) This function returns the height of the string str
when printed using the current font settings.

pr_print_text_block(x,y,str,w) This function also prints the string
str but within a block of width w. The string will be broken in multiple
lines, both at # and at carriage return characters, and such that the
width does not become larger than w. Use a negative w to not break
up lines.

pr_text_block_width(str,w) This function returns the actual width of the string str when printed using the block print function. (This can be smaller than the given width.)

pr_text_block_height(str,w) This function returns the height of the string str when printed using the block print function.

Printing Images

The extension package also allows for the printing of images. In particular there are functions for printing images from files, screenshots, sprites, backgrounds, and surfaces. However, realize that printing cannot handle the transparency of sprites and background resources. So if you want to print those you must give them a white background.

For files the following functions exist.

pr_print_file_simple(x,y,fname) Prints the image in the file fname at position (x,y). fname can be a .bmp or a .jpg file. Note that normally this will result in an image that is way too small. pr_print_file_scale(x,y,xs,ys,fname) Prints the image in the file fname at position (x,y) scaled horizontally with a factor xs and vertically with a factor ys. fname can be a .bmp or a .jpg file. pr_print_file_stretch(x0,y0,x1,y1,fname) Prints the image in the file fname stretched such that it fits in the region with corners (x0,y0) and (x1,y1) on the screen. fname can be a .bmp or a .jpg file. pr_print_file_part(xd0,yd0,xd1,yd1,xs0,ys0,xs1,ys1,fname) Prints the region with corners (xs0,ys0) and (xs1,ys1) from the image in the file fname stretched such that it fits in the region with corners (xd0,yd0) and (xd1,yd1) on the screen. fname can be a .bmp or a .jpg file.

For screenshots the following functions exist.

pr_print_screen_simple(x,y) Prints the screen at position (x,y). Note
that normally this will result in an image that is way too small.
pr_print_screen_scale(x,y,xs,ys) Prints the screen at position (x,y)
scaled horizontally with a factor xs and vertically with a factor ys.
pr_print_screen_stretch(x0,y0,x1,y1) Prints the screen stretched
such that it fits in the region with corners (x0,y0) and (x1,y1) on the
screen.

pr_print_screen_part(xd0,yd0,xd1,yd1,xs0,ys0,xs1,ys1) Prints the region with corners (xs0,ys0) and (xs1,ys1) from the screen stretched such that it fits in the region with corners (xd0,yd0) and (xd1,yd1) on the screen.

For sprites the following functions exist. Note that transparency is not taken into account here.

 $pr_print_sprite_simple(x,y,ind,subimg)$ Prints the subimage subing of the sprite ind at position (x,y). Note that normally this will result in an image that is way too small.

pr_print_sprite_scale(x,y,xs,ys,ind,subimg) Prints the subimage subing of the sprite ind at position (x,y) scaled horizontally with a factor xs and vertically with a factor ys.

pr_print_sprite_stretch(x0,y0,x1,y1,ind,subimg) Prints the subimage subing of the sprite ind stretched such that it fits in the region with corners (x0,y0) and (x1,y1) on the screen. pr_print_sprite_part(xd0,yd0,xd1,yd1,xs0,ys0,xs1,ys1,ind,subimg) Prints the region with corners (xs0,ys0) and (xs1,ys1) from the subimage subing of the sprite ind stretched such that it fits in the region with corners (xd0,yd0) and (xd1,yd1) on the screen.

For backgrounds the following functions exist. Note that transparency is not taken into account here.

 $pr_print_background_simple(x,y,ind)$ Prints the background ind at position (x,y). Note that normally this will result in an image that is way too small.

 $pr_print_background_scale(x,y,xs,ys,ind)$ Prints the background ind at position (x,y) scaled horizontally with a factor xs and vertically with a factor ys.

pr_print_background_stretch(x0,y0,x1,y1,ind) Prints the background ind stretched such that it fits in the region with corners (x0,y0) and (x1,y1) on the screen.

pr_print_background_part(xd0, yd0, xd1, yd1, xs0, ys0, xs1, ys1, ind)
Prints the region with corners (xs0,ys0) and (xs1,ys1) from the background ind stretched such that it fits in the region with corners (xd0,yd0) and (xd1,yd1) on the screen.

For surfaces the following functions exist.

 $pr_print_surface_simple(x,y,surf)$ Prints the surface surf at position (x,y). Note that normally this will result in an image that is way too small.

 $pr_print_surface_scale(x,y,xs,ys,surf)$ Prints the surface surf at position (x,y) scaled horizontally with a factor xs and vertically with a factor ys.

pr_print_surface_stretch(x0,y0,x1,y1,surf) Prints the surface surf stretched such that it fits in the region with corners (x0,y0) and (x1,y1) on the screen.

pr_print_surface_part(xd0,yd0,xd1,yd1,xs0,ys0,xs1,ys1,surf) Prints the region with corners (xs0,ys0) and (xs1,ys1) from the surface surf stretched such that it fits in the region with corners (xd0,yd0) and (xd1,yd1) on the screen.

Printer Dialogs

Printing normally happens on the default printer. To let the user change the default printer you can call the following function to open the printer setup dialog:

pr_printer_setup_dialog_show() This function shows the printer setup dialog in which the user can indicate options for the printer. It returns whether the user pressed **Ok** although nothing really needs to be done with that. The settings are automatically applied.

There is a second dialog available to let the user indicate print options.

pr_print_dialog_show() This function shows the print dialog in which the user can indicate the printer to use and the print options. It returns whether the user pressed Ok.

This dialog will allows the user to set a number of options. First of all the user can set the printer. This is handled automatically. Secondly, the user can indicate how many copies to print and whether to collate them. There are a number of functions to set and get information about this but it is again handled automatically by the system.

pr_print_dialog_set_collate(collate) Set whether to collate the pages.

pr_print_dialog_get_collate() Returns whether to collate the pages. (The system will automatically handle this.)

pr_print_dialog_set_copies(copies) Sets the number of copies.
pr_print_dialog_get_copies() Returns the number of copies. (The system will automatically handle this.)

Also the user can indicate which pages to print, if you allow this in the dialog. There are a number of functions for this. Note that this is **not** automatically handled by the system. You must print the appropriate pages and not the other ones.

pr_print_dialog_set_frompage(numb) Sets the first page (when a range is given).

pr_print_dialog_get_frompage() Returns the first page (when a range is given).

pr_print_dialog_set_topage(numb) Sets the last page (when a range is given).

pr_print_dialog_get_topage() Returns the last page (when a range is given).

pr_print_dialog_set_minpage(numb) Sets the smallest page number
that the user is allowed to indicate (when a range is given).
pr_print_dialog_get_minpage() Returns the smallest page number
that the user is allowed to indicate (when a range is given).
pr_print_dialog_set_maxpage(numb) Sets the largest page number
that the user is allowed to indicate (when a range is given).
pr_print_dialog_get_maxpage() Returns the largest page number
that the user is allowed to indicate (when a range is given).
pr_print_dialog_set_range(range) With this function you can
indicate the type of page range for printing. The following options
exist:

pr_range_all = all pages must be printed pr_range_selection =
the selection must be printed pr_range_pages = the indicated
page range must be printed

pr_print_dialog_get_range() Returns the page range, using the same values as above.

Finally there some dialog options you can set.

pr_print_dialog_set_option(option, value) With this function different options can be set. Each option has an index and the value should be true or false. The following options exist:

pr_printtofile = whether print to file option must be shown
pr_disableprinttofile = whether print to file option must be
disabled pr_pagenums = whether the user can select a range of
pages pr_selection = whether the user can indicate that a
selection must be printed pr_warning = whether a warning must

be giving when the printer does not exists

pr_print_dialog_get_option(option) With this function you can check the value of the different options. You can use the same options as indicated above.

Direct Printer Selection

If you do not want to use the dialogs described above you can also do the printer selection yourself or create a room in which the player can do it. Each printer has an index and a name. The following functions exist to receive the index and name of the current printer and to set the printer to be used.

pr_get_printer() Returns the index of the currently selected printer.
pr_get_printer_current() Returns the name of the currently
selected printer.

pr_get_printer_count() Returns the number of printers. The printers
will be number from 0 till this number minus one.

pr_get_printer_name(ind) Returns the name of the printer with index
ind.

pr_set_printer(ind) Sets the printer to be used to be the printer with
index ind.

pr_set_printer_default() Sets the printer to be used to the default
printer. Use this before calling pr_begin_document().