ATEX
extension

To initialize extension write
ts_init()
before using!

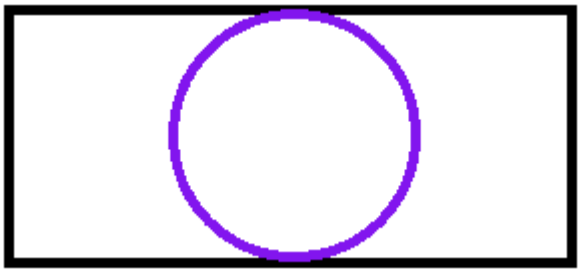
Atex using tags to modify text. Just add tag to string with ts_... functions. Ex:
txt='Good '+ts_font(f_italic)+'morning'

You can combine any tags without limits, ex:
txt='I am so '+ts_shake(1)+ts_colour(c_red)+'angry!'

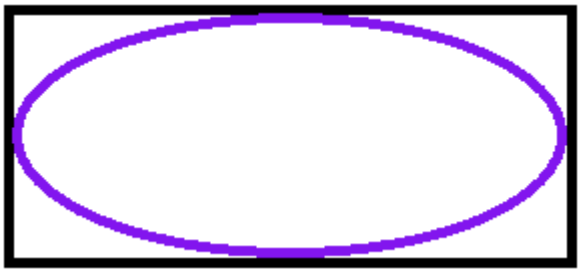
Supported tags:

- *ts_image(sprite_index, [w, h, type, [x, y]])*

sprite_index - index of drawing sprite
To use image like symbol skip w and h arguments or set it to ts.standart or -1
To use original size of image set w and h to ts.original or -2
Or input w and h in pixels to use your own size
- type - type of scaling:
  - ts.img_scale - image scaling to current size and saving proportions

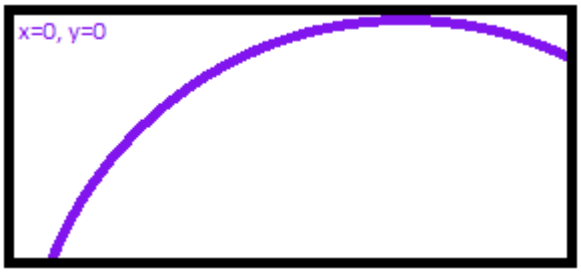


- ts.img_fullscale - image scaling without saving proportions



- ts.img_partscale - image scaling, save proportions and unnecessary parts are cut

- ts.img_part - drawing part of image without scaling, use arguments "x" and "y" to set what part will be draw, or skip to default value(0)



- *ts_line([sep])*

Break line with current distance

- *ts_lineback()*

Break line without distance

- *ts_font(newfont)*

Set font of next text

- *ts_outline(bool, [colour])*

Set text outline

- bool - enable, true or false
- colour - colour of outline, skip to use colour of text

- *ts_underline([width, colour])*

Set text underline

- width - width of underline, skip or set 0 to off underline
- colour - colour of line, use ts.standart or -1 to set color of text

- *ts_shadow(bool, [pos_x, pos_y, colour, alpha])*

Set shadow of next text

- bool - enable, true or false
- pos_x and pos_y - relative position of shadow
- colour - colour of the shadow
- alpha - alpha of the shadow

- *ts_link(url_or_script, colour_moveon, colour_press)*

Start of link, to end link use ts_link() without any argument

- url_or_script - url or index or name of script to execute
- colour_moveon - colour of text when mouse enter
- colour_press - colour of text when user press on link

Ex:

txt='please, press on '+ts_link('https://yoyogames.com', c_green, c_red)+'me'+ts_link()+' to open yoyogames site'

- *ts_halign(halign)*

Set halign of text on current and next lines

- *ts_valign(valign)*

Set valign of text on current and next lines

Examples of use halign and valign tags you can find in ATEX build-in tutorial

- *ts_colour(colour)*

Set colour of next text

- *ts_space(space_w)*

Create empty space with current width

- *ts_glow(colour)*

Set GLOW effect for text with current colour

!!! Next tags work only when ts_effects(true) is written !!!

To save FPS, write ts_effects(false) when you didn't need in effects

- *ts_shake([strong])*

Set shake effect, set strong to 0 to off

- *ts_wave([wave_spd, wave_strong])*

Set wave effect to next text

  - wave_spd - speed of wave, 1 - slow, 10 - fast and etc...
  - wave_strong - max distance in pixels

- *ts_drawscript([script, index])*

Set draw script. With this tag you can create your own effects.

- script - script that will draw anything, use -1 to disable script on current index
- index - index of script(work like depth), from -15 to -1 - executing before drawing text, 0 - executing instead standart drawing, from 1 to 15 - executing after drawing text

example of script whose doing nothing:

draw_text(argument0, argument1, argument2)

example of a script that reverses the text:

draw_text_transform(argument0, argument1, argument2, 1, -1, 0)

You can change ts_current_symbol and ts_current_x/ts_current_y in your script to change position and symbol

-draw_text_special

This is the most convenient way to draw formatted text. Just write draw_text_special(...) and it will be drawn

draw_text_special(x, y, text, [halign, valign, font, w, count])

- x and y - pos of text
- text - text to draw
- halign and valign - global align of text, you can use fa_center, fa_left, fa_top and etc...
- font - standard font
- w - max width of text, use -1 or ts.standart or skip to draw without limits
- count - count of symbols to draw, use -1 or skip to draw all symbols

This function support all tags, but it is slow to draw big texts

-text_format

This is second way to draw formatted text. It faster than draw_text_special, but you should to use text_format to get id of formatted text and after it draw using text_format_draw. You can use text_format in create and text_format_draw in draw event

text_format(text, [font, w])

- text - text to format
- font - standart font
- w - max width

it returns id of formatted text

text_format_draw(x, y, text_formatted, halign, valign, font[, count])

- x, y - pos of text
- text_formatted - id of formatted text
- halign and valign - global align
- font - standart font
- count - count of symbols to draw

if you use count, it returns true if count>=count of symbols in text

text_format_destroy(text_id)

- text_id - id of formatted text

this function cleans up the memory

-Static text

This is third way to draw formatted text. This is fastest way, but dynamic tags like links, shake and wave work incorrect

To create static text use text_create(...), to draw static text use text_render(...)

text_create(text, font, maxw)

- text - your text
- font - standart font
- maxw - max width, use -1 to draw without limits

This function return id of static text

text_refresh(text_id, font, maxw)

- text_id - id of static text
- font - standart font
- maxw - max width

This function refresh static text. You shouldn't use it because it refreshed automatically (But you can. It is documentation, not a cop)

text_set_text(text_id, newtext, font, maxw)

- text_id - id of static text
- newtext - text to set
- font - standart font
- maxw - max width

This function set new text in static text

text_render(text_id, x, y, halign, valign)

- text_id - id of static text
- x and y - pos of text
- halign, valign - global align of text

This function draw static text

text_destroy(text_id)

This function clear up memory

Advanced using:

text_create_sprite(text_id, smooth, xorig, yorig)

- text_id - id of static text to create sprite
- smooth - enable smooth
- xorig, yorig - orig of sprite

This function create and return sprite with current static text

text_get_surface(text_id)

- text_id - id of static text

This function return surface of static text. It does not re-create it

text_width/height(text_id)

Current functions return width and height of static text

This method is very fast and it is created to draw large static texts. These functions control the surface by themselves, you can not be afraid that the surface will be destroyed

Keywords

To easy text creating you can use keywords: it key-words that change themselves to tags or parts of text.

ts_add_keyword(key, tag)

- key - key-word
- tag - tag or part of text to change

This function creates keyword

For example:

ts_add_keyword('$red#', ts_colour(c_red))
txt='this colour is $red#red'

Result:

this colour is red

Another example:

global.name=ts_colour(c_red)+'Cherry'+ts_colour(c_black)
ts_add_keyword('[name]', global.name)
txt='Hello, [name], how are you?'

Result:

Hello, Cherry, how are you?

ts_replace_keyword(key, newtag)

- key - key-word
- newtag - new tag to change

This function changes tag to newtag in current keyword

Usertags

You can use usertags to create your own tags(based in existing tags)

usertag_enable(bool)

- bool - true/false

This function set on/off usertags

usertag_add(start_trigger, argument_count, end, script)

- start_trigger - trigger of tag. Ex: “<tag” or “[tag” or “tag” and etc...
- argument_count - count of arguments of current tag. You can set it to -1 to get unlimited arguments
- end - end trigger. Ex: “>” or “end>” or “end” and etc...
- script - script of usertag

This function create usertag

Example:

usertag_enable(true)
usertag_add(‘<font=’, 1, ‘>’, scr_changefont)
txt=’<font=f_norm|>Hello <font=f_big|>World!’

scr_changefont:

return ts_font(argument0[| 0])

Result:

Hello World!

Ever tag argument MUST end with a “|” character
All arguments wrote by user will be in ds_list in argument0

Additional opportunity

ts_delete_lastchar(string)

- string - your text

Delete last char in text (tags, usertags and keywords will be deleted as 1 symbol)

ts_constant(name, value)

- name - name of constant
- value - returned value

Create constant. You can use it in usertags or in tags as argument

ts_justify(enable)

- enable - true/false

Set justify for text

ts_effects(enable)

- ebanle - true/false

enable/disable effects(shake, wave and scripts)

ts_wordwrapping(enable)

- true or false

enable/disable word wrapping

END

I hope that my extension will be useful to you

About bugs, alerts and your ideas write in MrSanyaShanin@yandex.ru or in my vk profile

Page of extension - https://marketplace.yoyogames.com/assets/6533/atex

Local chat based on ATEX - https://marketplace.yoyogames.com/assets/6551/local-chat

All my extensions - https://marketplace.yoyogames.com/publishers/2522/vishnya-games

If you have a lot of extra money and you want to donate me, use my Viki wallet: 89603497405

Thanks :^)