Lua Graphics - Notifications Documentation

Overview

As you might know already, our project have an in-built notifications system. A notification is basically a box that spawns in a given position, containing some information. The cool part about them is that the user can interact with them, allowing you to create interactive functions. For example, if you are a hunter and your pet dies, you can send a notification that warns the user that the pet has died. Since the notifications are interactive, as previously said, you could add a functionality to revive the pet if the notification is clicked.

Basic Functionality Explanation

Using notifications is very simple, since almost everything is handled internally. You just have to keep in mind a couple key points:

1- Callbacks: You can use all notifications functionalities from any callback, since the rendering is handled internally.

2- Positioning: By default, all notifications are rendered in the position that is specified in the main menu (System -> Notifications). However, you can still customize their spawn position, although this is not recommended in general since the user might be expecting all notifications from all plugins to spawn in the same place. You can still do something like the Hunter Plugins notifications customizations, where by default the position is the same as the main menu one, but the user can specifically customize the notifications from your plugin.

3- Identification: Every notification must have its ***own unique ID***, same like with menu elements. This ID is a string, so we recommend using local variables (defined outside of the callbacks) that are easy to recognize for each individual notification. Only 1 notification with the same ID can be active at a time.

Functions 🛠️

Add Notification 🔔

Syntax

```
core.graphics.add_notification(header, message, duration_s, color, x_pos_offset, y_pos_offset, max_background_a
```

Parameters

- `header` : `string` - The information text for the notification that will appear on top.
- `message` : `string` - The message text for the notification. This is the actual notification information.
- `duration_s` : `integer` - The duration of the notification in seconds.
- `color` : color - The color of the notification.
- `x_pos_offset` (Optional): `number` - The x-position offset for the notification. Default is `0.0`.
- `y_pos_offset` (Optional): `number` - The y-position offset for the notification. Default is `0.0`.
- `max_background_alpha` (Optional): `number` - The maximum background alpha value. Default is `0.95`.
- `length` (Optional): `number` - The length offset of the notification (This value adds up to the default notification length). Default is `0.0`.
- `height` (Optional): `number` - The height offset of the notification (This value adds up to the default notification height). Default is `0.0`.

Description

Adds a notification with the specified information, message, duration, color, and optional positional offsets, background alpha, length, and height.

Example:

Adding a notification after right mouse button was clicked

```lua
---@type color
local color = require("common/color")

local notification_id = "rmb_pressed_notification"

local function notify_rmb_was_pressed()
    -- you can avoid this check if you checked it earlier in your code.
    -- It's just to make sure nothing is rendered while on loading screen.
    local local_player = core.object_manager.get_local_player()
    if not local_player then
        return nil
    end

    local is_rmb_pressed = core.input.is_key_pressed(0x02)

    if is_rmb_pressed then
        core.graphics.add_notification(notification_id, "[Notifying]", "RMB Was Pressed!", 5, color.get_rainbow
    end
end

```

Is Notification Clicked 🔔🖱️

Syntax

```
core.graphics.is_notification_clicked(id, trigger_after_time)
```

Parameters

- `id` : `string` - The ID of the notification to check.
- `trigger_after_time` (Optional): `number` - The time in seconds after which the notification click is triggered. Default is `0.0`.

Returns

- `boolean` : `true` if the notification has been clicked, `false` otherwise.

Description

Checks if a notification with the specified message has been clicked, with an optional trigger time delay.

Get Notifications Core Position 📍

Syntax

```
core.graphics.get_notifications_core_pos()
```

Returns

- vec2: The core position of the notifications.

Description

Retrieves the core position of the notifications. This is the position that can be customized in the main menu (System -> Notifications)

Get Notifications Default Size 📏

Syntax

```
core.graphics.get_notifications_default_size()
```

Returns

- vec2: The default size of the notifications.

Description

Retrieves the default size of the notifications. This size cannot be modified by user input.

Complete Example

Lets finish off with an example that summarizes all functionality. The code will add a notification when RMB is pressed by the user. It will spam in the console whether the notification is active or not, and if it's clicked by the user, it will print so in the console and the notification won't be shown again until the LUA modules are reloaded.

Summarizing Example:

Interiorizing the concepts

```lua
---@type color
local color = require("common/color")

local notification_id = "rmb_pressed_notification"

local function notify_rmb_was_pressed()
    -- you can avoid this check if you checked it earlier in your code.
    -- It's just to make sure nothing is rendered while on loading screen.
    local local_player = core.object_manager.get_local_player()
    if not local_player then
        return nil
    end

    local is_rmb_pressed = core.input.is_key_pressed(0x02)

    if is_rmb_pressed then
        core.graphics.add_notification(notification_id, "[Notifying]", "RMB Was Pressed!", 5, color.get_rainbow
    end
end

local notification_ended = false
core.register_on_update_callback(function()
    if notification_ended then
        return
    end

    notify_rmb_was_pressed()

    local is_notification_clicked = core.graphics.is_notification_clicked(notification_id)

    if not is_notification_clicked then
        core.log("Is Notification Appearing On Screen: " .. tostring(core.graphics.is_notification_active(notif
    else
        core.log("Is Notification Clicked: " .. tostring(is_notification_clicked))
        notification_ended = true
    end
end)
```