Beginner BLT modding

mod.txt

Every mod needs this file. This file is responsible for loading your mod with BLT. Below you will find the `mod.txt` used for this example. Copy the contents inside your own `mod.txt`. Most of the fields should be self explanatory expect for the `hooks`.

```
{
	"name": "Hide loud jobs",
	"description": "Hides jobs from crime.net which are not stealthable",
	"author": "Your name",
	"version": "1.0",
	"blt_version": 2,
	"hooks": [
		{
			"hook_id": "lib/managers/crimenetmanager",
			"script_path": "mod.lua"
		}
	]
}
```

Hooks

- `hook_id` is the path to the file (from the original codebase) in which the function is defined which we want to manipulate.
- `script_path` is the path to the file in your mods folder where your code is in.

Finding the code

There's no magic trick to finding the right place in the source code. It takes practice and understanding of the codebase. We will not be covering that here but you are more than welcome to ask the community for help in the `#payday-help`-channel of our Discord on this topic.

You can still follow along by opening the file `lib/managers/crimenetmanager` (this is also the `hook_id`) inside the source code you downloaded earlier. Find the function `CrimeNetGui:add_server_job` (or view it here).

Editing the code

Copy the entire function in your `mod.lua`. Now we will add an `if`-statement around the code inside the function. This statement will check if it is indeed a stealthable mission. Only if that's the case will the game actually add it to the map. Luckily there is already such a function built into the game (as it's used to display the little ghost icon next to the job). You can find that part in the same file (or view it here). Let's have a look at it:

```
if data.job_id and managers.job:is_job_ghostable(data.job_id) then
	ghost_icon = icon_panel:bitmap({
		texture = "guis/textures/pd2/cn_minighost",
		name = "ghost_icon",
		blend_mode = "add",
		color = tweak_data.screen_colors.ghost_color
	})

	ghost_icon:set_top(side_icons_top)
	ghost_icon:set_right(next_icon_right)

	next_icon_right = next_icon_right - 12
end
```

From simply reading the code: If the `job_id` exists and the job with the given `job_id` is stealthable, we create a new icon with the texture of the blue ghost and set its position. Now let's try to apply that to our code. Our function is also given the parameter `data`. This is in fact the same object for this case. Therefore we can safely wrap our code in the exact same `if`-statement like this:

mod.lua

Here's how your finished mod.lua should look like:

```
function CrimeNetGui:add_server_job(data)
	if data.job_id and managers.job:is_job_ghostable(data.job_id) then
		local gui_data = self:_create_job_gui(data, "server")
```

```
    gui_data.server = true
    gui_data.host_name = data.host_name
    self._jobs[data.id] = gui_data
  end
end
```

That's it - Try it out!

► End of basic tutorial ◄

Continuation: Finding the code

I will explain to you my thought process of how I was able to find the correct function in case it may help you:

For this example I was specifically looking for `crimenet` which brought me directly to the `CrimeNetManager`. I knew that stealth is mostly referred to as `ghost` in the code - so I quickly picked up on the idea of the `ghost_icon` which was used in the function `_create_job_gui` but that one is difficult to grasp and gets called through the more accessible function `add_server_job`. Now I only had to go back to the `ghost_icon` to figure out how the game determined if it should be displayed or not.

Different methods:

Let's review some other methods we could have used to get this mod working.

[Hard Overwrite] (as shown above)

- PRO: You are in full control by modifying the original source code.
- CON: Only 1 mod installed can hard-overwrite the function (not necessarily yours). All other mods manipulating the same function will not work.

```
function CrimeNetGui:add_server_job(data)
  if data.job_id and managers.job:is_job_ghostable(data.job_id) then
    local gui_data = self:_create_job_gui(data, "server")
```

```
    gui_data.server = true
    gui_data.host_name = data.host_name
    self._jobs[data.id] = gui_data
  end
end
```

Explanation:

This method is the most straightforward: It is the exact copy from the source code but with our adjustments.

[Soft Overwrite]

- PRO: good for functions which return values.
- CON: Still issues which compatibility as functions are chained/nested inside each other and depending on loading order other mods could alter that behavior.

```
local old_add_server_job = CrimeNetGui.add_server_job


function CrimeNetGui:add_server_job(data)
  if data.job_id and managers.job:is_job_ghostable(data.job_id) then
    old_add_server_job(self, data)
  end
end
```

Explanation:

This method works by keeping a copy to the original function. This function is then overwritten with our implementation where we call the reference to the original function.

[Hook]

- PRO: The original function stays completely untouched.
- CON: Does not work for functions which return values. We will cover techniques to circumvent these issues in the "Advanced BLT modding".

```
Hooks:PostHook(CrimeNetGui, 'add_server_job', 'hideloudjobs_crimenetgui',
function(self, data)
  if data.job_id and not managers.job:is_job_ghostable(data.job_id) then
    self:remove_job(data.id, true)
  end
```

```
end)
```

Explanation:

Hooks don't touch the original function. They get executed just **before** (`Pre`) or **after** (`Post`) the function.

So which one should you use?

Generally speaking: Hooks > Soft Overwrite > Hard Overwrite

Use Hooks wherever you can. Use Soft Overwrites when dealing with function where you need to change the returning value and only use the Hard Overwrite method if you really have to.

Revision #14
Created 8 August 2020 18:14:52 by The_Punisher
Updated 9 July 2022 18:34:54 by The_Punisher