Menu

19/8/17

When the game starts up, it shows the menu screen. From here, the player can choose story or endless mode, or go to the settings menu. There will be:

* Main screen
* Level select screen (in story mode)
* Settings screen

Therefore, there are three menu states.

The menus will be in their own loop; all the normal game processing will not be completed. However, should the drawing code be the same? The groups should not be defined yet, so the drawing (and all processing for that matter) should be entirely separate from the rest of the game code.

How do you separate the game code with menu code? I think catching the menus in a loop is a bad idea – it would make it difficult to add anything on later. That said, I can’t think of anything I would add. The game loop should be call-able – not global, as it is now. I’d just feel a lot better about it that way.

Like what you did at work, have the main loop in a state machine. In the while 1, depending on the state, it calls a function – menu, game, whatever else (credits, etc). Don’t use a switch or if/else for every frame, try and point to the function please.

Game loop state machine – not to be confused with menu state machine

|  |  |  |
| --- | --- | --- |
| State | Purpose | Notes |
| 0 | Initialise the main menu | Eventually, you can replace with splash screens? Splash screens suck actually, so nvm |
| 1 | Menu loop | Loop the menu, check music loop, check for player inputs and that |
| 2 | Initialise game | Depending on menu selection parameters and save data |
| 3 | Game loop | Just the game |

When the game starts up, do the screen setup and definitions for state 0.

When that is completed, return a flag to put the game into the menu loop.

In the menu, loop until flag is returned to enter into state 2, to initialise the game.

Once the game is initialised (groups are made, difficulty level set, level defined), enter the game loop. If any flag to change state is given, un\_init the game and go to that state. Does there need to be an un\_init state?

How do I define something in one function then go out and into another function with those definitions still in place? I don’t think you can. This idea is really just complicating things – stick to the main script state machine, but have the loops internal to each state.

21/8/2017

ATM There’s a big loop that holds ifs for the menu and the game. It goes to the menu – if the right buttons are pressed, it sets the startGame flag and exits the menu loop. Then it enters the game script, which has its own loop. Upon quitting that, it goes back to the main script, and loops back into the menu. From the menu, you can quit the game.

The next job is to ~~meme it up~~ add options for the menu – stuff like game start, level selct, options, quit game. For each sub-menu, there will be multiple positions the cursor can be in. When you press the move keys, it changes the cursorState depending on what menu we’re in, and what direction it should move to. There will be a function to take the input and determine the new state. That could be a new cursor state or a new menu state.

When entering a new menu, the cursor state will always be set to 0, the first state/option of that menu. So, that much is easy, innit. The first thing will be to check if the direction is in/out – to move up a menu or down. If so, depending on the current menu, go to the new menu. That just means change menu state; in the computation part, it will remove all objects/texts from the drawGroups, and add the appropriate new ones. If a new direction is input, just change cursor state- the computation part will set the new position using lerping or something similar.

Instead of if elses for days, make a table that has menu state and cursor state, and returns a value depending on the input. I think it’s best there are separate functions for moving the cursor and the menu, we never do both at once.

Need to differentiate between what happens upon menu change and what is gradual. For example, screen transitions; when a state is changed, set some variables for your fancy lerp biz. Also, the current function only returns the new state – should it also do stuff like change to the game or exit? I think upon state change there should be a function that checks for special events like this.

Making the cursor move to its option – as all the menu options are in a list, we can just get the position of object whose index matches the cursorPos.

27/10/17

The menu goes between menus just fine, but what about actually doing stuff, like changing options?

To activate, enter must be pressed, that’s the trigger. Then, it must check, from the states, what the action should be – specific function, or generic menu change. Treat it like a switch; check if the state is a special state, otherwise default to change menu.