Odd Semester (2021)



**BINUS UNIVERSITY**

**BINUS INTERNATIONAL**

**Assignment Cover Letter**

**(Individual Work****)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | |  | |  | |
| **Student Information**: **Surname** | | | | | **Given Names**  **Georgius** | | **Student ID Number**  **2101704571** | |
| 1. | | **Kurli** |  | |
|  |  |
| **Course Code** | **: COMP6502** |  |  | | **Course Name** | | **: Introduction to Programming** | |
| **Class** | **: L1AC** |  |  | | **Name of Lecturer(s)** | | **:** 1. IDA BAGUS KERTHYAYANA | |
|  |  |  |  | |  | | 2. TRI ASIH BUDIONO | |
| **Major** | **: CS** |  |  | |  | |  | |
| **Title of Assignment**  (if any) | : Tree Cutting Simulation | |  |  | |  | |  | |
| **Type of Assignment**    **Submission Pattern** | **: Final Project** |  |  | |  | |  | |
| **Due Date** | **: 7-11-2017** |  |  | | **Submission Date** | | **:** | |

The assignment should meet the below requirements.

1. Assignment (hard copy) is required to be submitted on clean paper, and (soft copy) as per lecturer’s instructions.
2. Soft copy assignment also requires the signed (hardcopy) submission of this form, which automatically validates the softcopy submission.
3. The above information is complete and legible.
4. Compiled pages are firmly stapled.
5. Assignment has been copied (soft copy and hard copy) for each student ahead of the submission.

# Plagiarism/Cheating

BiNus International seriously regards all forms of plagiarism, cheating and collusion as academic offenses which may result in severe penalties, including loss/drop of marks, course/class discontinuity and other possible penalties executed by the university. Please refer to the related course syllabus for further information.

# Declaration of Originality

By signing this assignment, I understand, accept and consent to BiNus International terms and policy on plagiarism. Herewith I declare that the work contained in this assignment is my own work and has not been submitted for the use of assessment in another course or class, except where this has been notified and accepted in advance.

Signature of Student: (Name of Student)

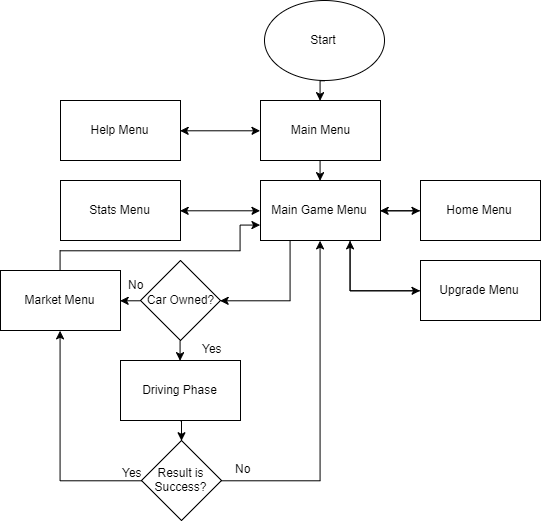
1. Georgius Easton Kuri

Tree Cutting Simulation Game

Introduction

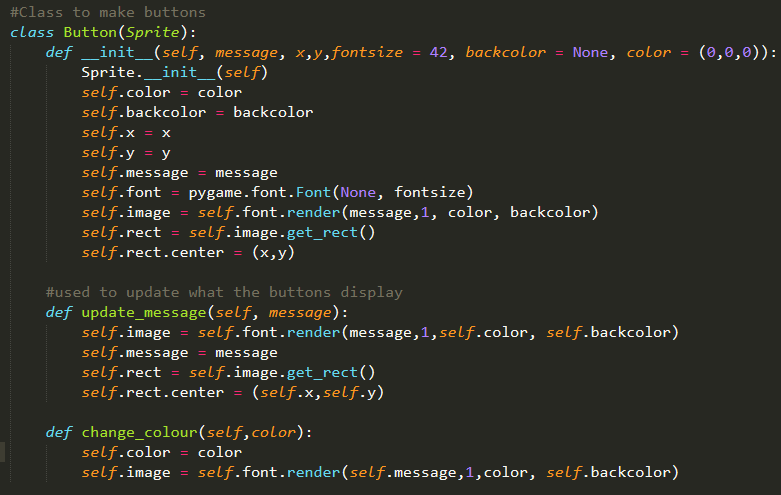
This program is an arcade-like game that focuses on earning money from selling logs. The purpose of this program is to entertain people who feels bored.

Flowchart



Explanation of Classes in Game\_Classes.py:

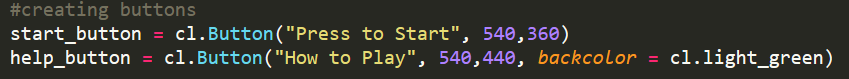
Button Class

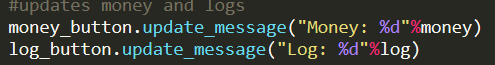


This class is used to make buttons that require text to be printed.

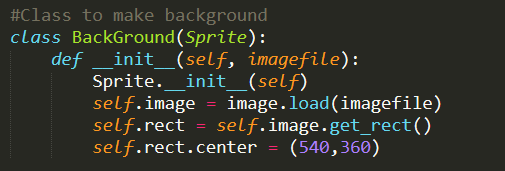
The class has default values such as font size 42, no backcolor and white font color. The class has a function “update\_message” that updates the object with a new message. The function “change\_colour” is used to change the text colour.

Example:





BackGround Class

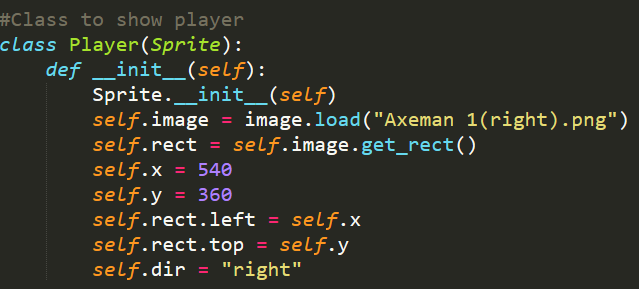


This class is used to create a background sprite at the center of the screen.

Example:



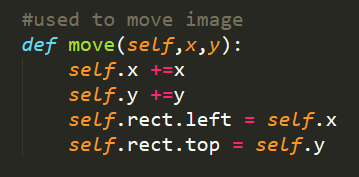
Player Class



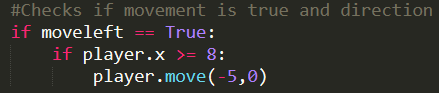
This class is used to create the player.

The player starts at the center of the screen facing the right.

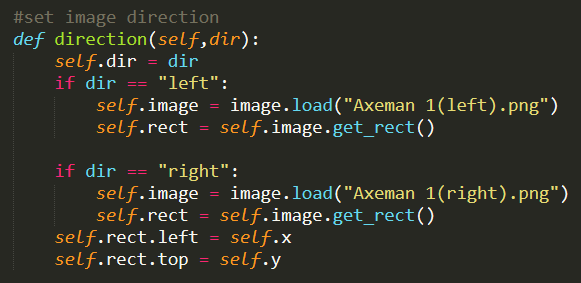
Move function in Player class



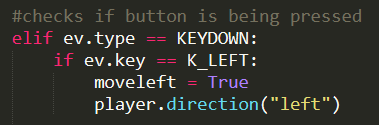
This function is used to move the player image up, down, left or right.

Example: 

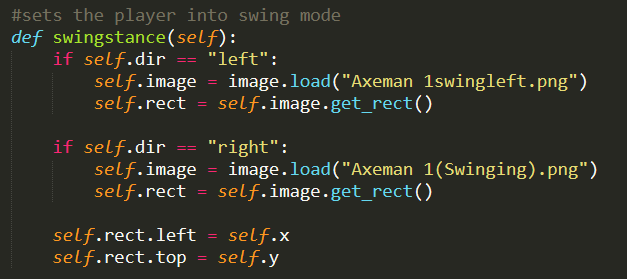
Direction function in Player class



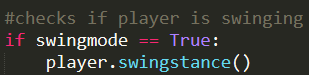
This function is used to check the player’s direction when moving and to change where the player is facing.

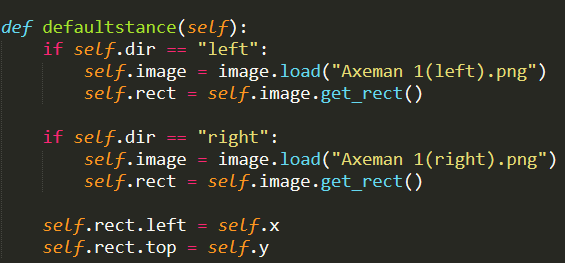
Example: 

Swingstance function in Player class

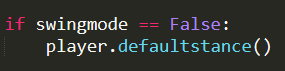


This function is used to set the player into swing mode by changing the character image. It also takes player direction into account.

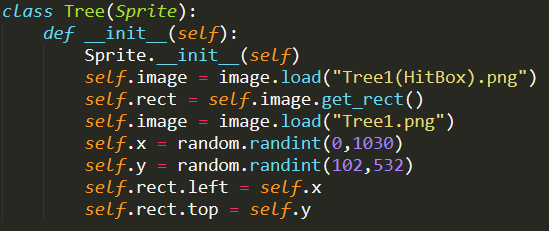
Example:

Defaultstance function in player class

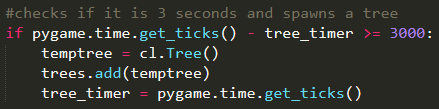
This function is used to set the player back into idle mode.

Example:

Tree Class



This class is used to create the tree sprites. It spawns the tree in a range of coordinates using randomint.

Example:

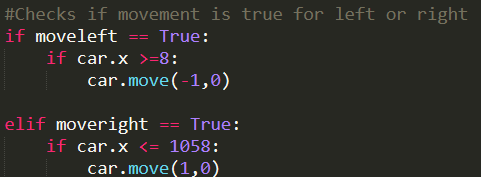
Car Class



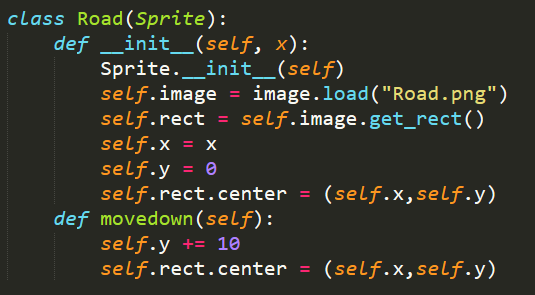
This class is used to change the player into a car in driving phase.

It has the function “move” that is used to move the car image.

Example of Car object:

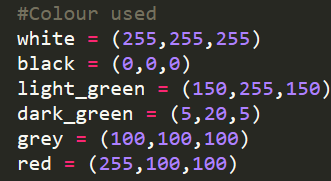
Example of move function:

Class Road



This class is used to create roads for the driving phase. The function “movedown” is used to move the road down.

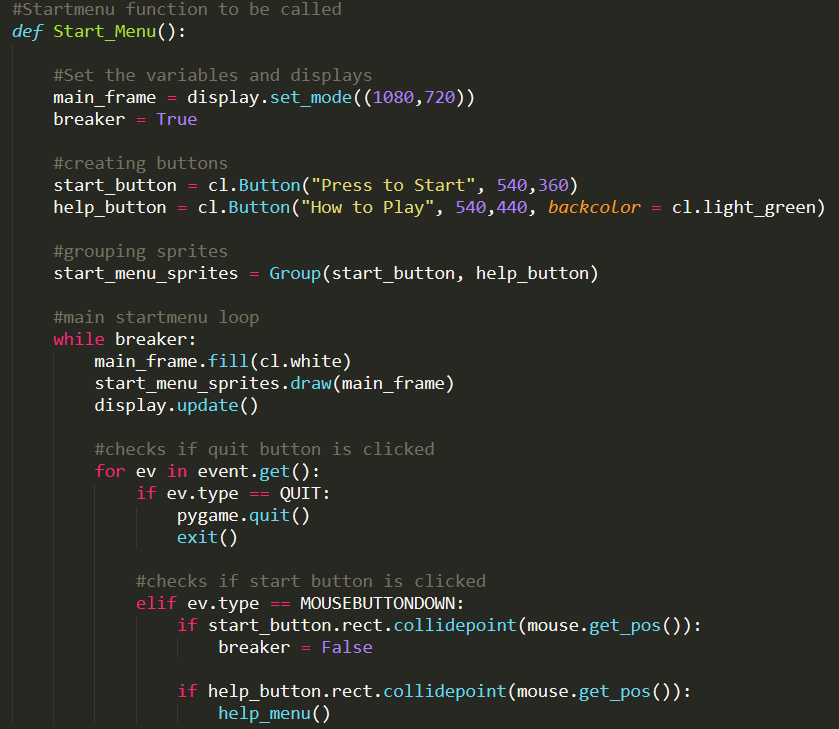
Global Variables in Game\_Classes



These are the colours used in the program.

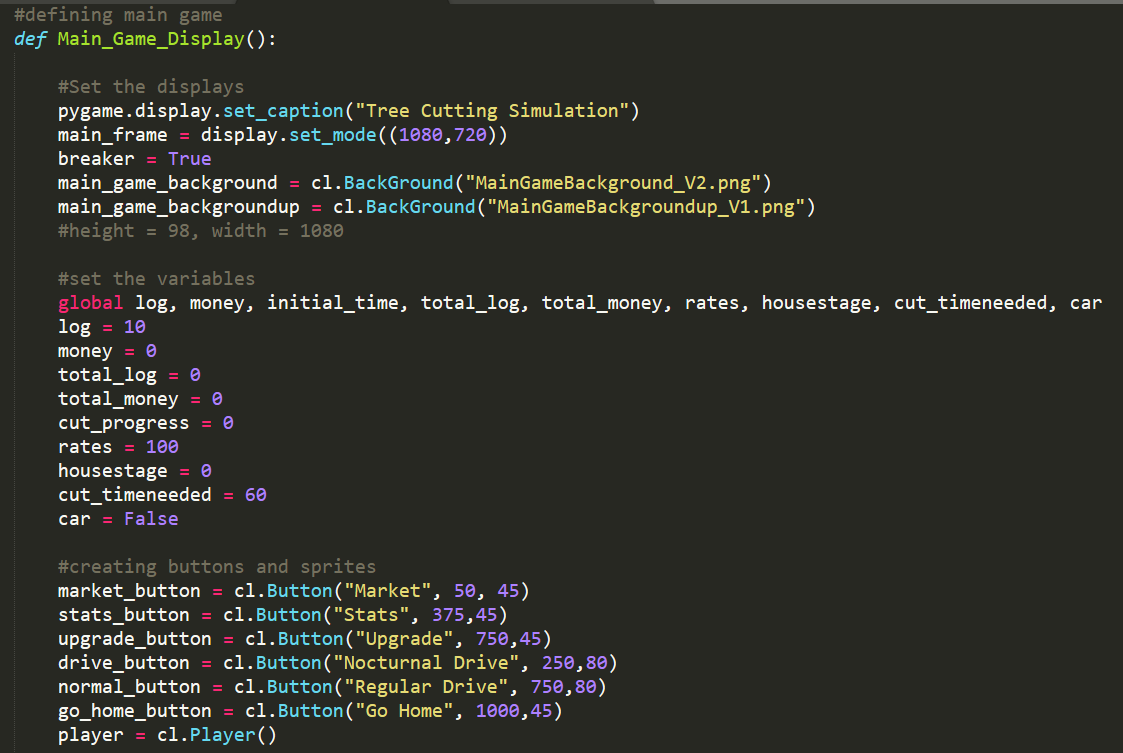
Explanation of Functions in Game\_Phase.py

Function Start\_Menu()

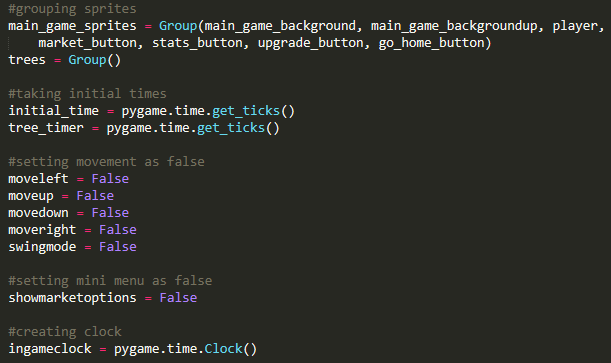


This function is used to show the starting menu for the game. It has two buttons. One is to go into the help menu and the other is to start the game. If the start button is pressed, it will break the menu and move on to the Main\_Game\_Display().

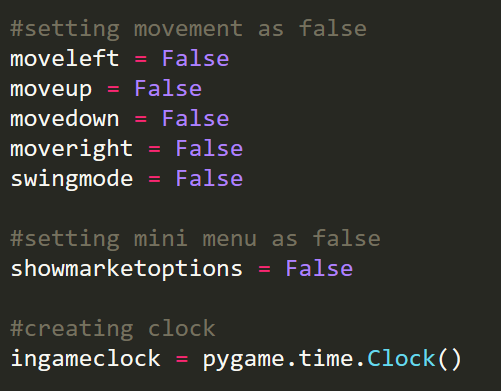
Function Main\_Game\_Display()



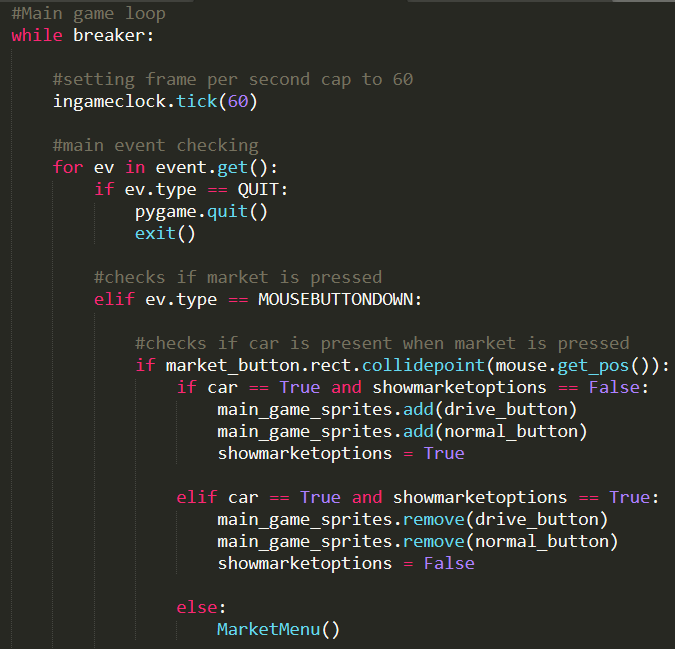
This function is used as the main game. It acts as both a menu and game. Variables, backgrounds, buttons and other objects are set first.



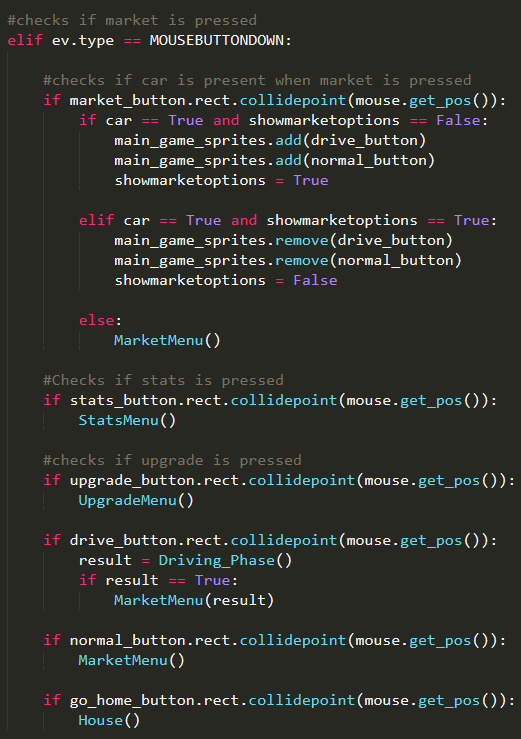
This part is used to create a group for the trees, buttons, backgrounds and the player. Initial\_time and tree\_timer is set as well.



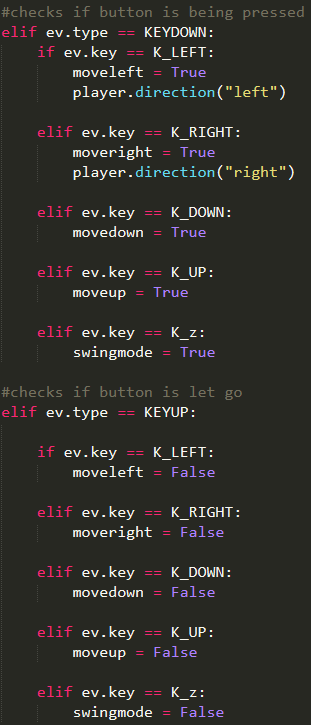
This part is used to set all variables used to check input to false. It also sets the ingameclock as a Clock().



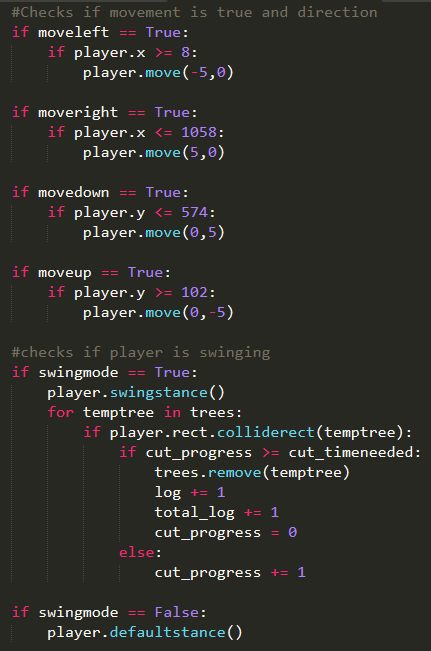
This is the main loop of the function. Ingameclock.tick(60) is used to limit the frame per second to 60. The for loop is used as an event checker. The first event checked is if the event is equal to QUIT, the program will close.



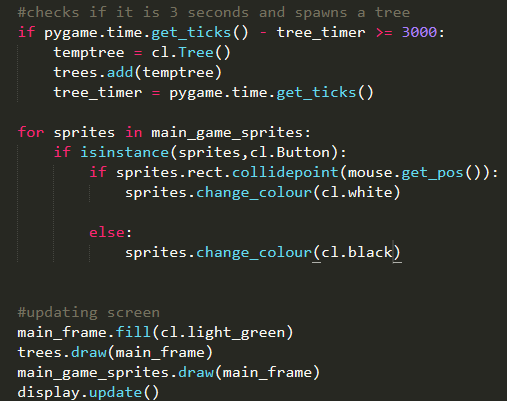
This part is used to check for mousebutton events. If a mousebutton event is inputted, it will check if the mouse is on any of the buttons and will run what the button is intended to do. For example, if the upgrade button is clicked on, it will run UpgradeMenu() which is responsible for the upgrade display. For the market, the program will check first if the car variable is True. If it is true, then it will show more options such as nocturnal drive or regular drive. If it is false, it will just run MarketMenu().



This part checks if any of the arrow keys or the z key are pressed. If they are pressed, it will set the respective variables as True. It also makes it so that the character faces left or right respective to the key pressed. If any of the arrow keys or the z key is let go, the respective variables will turn False.

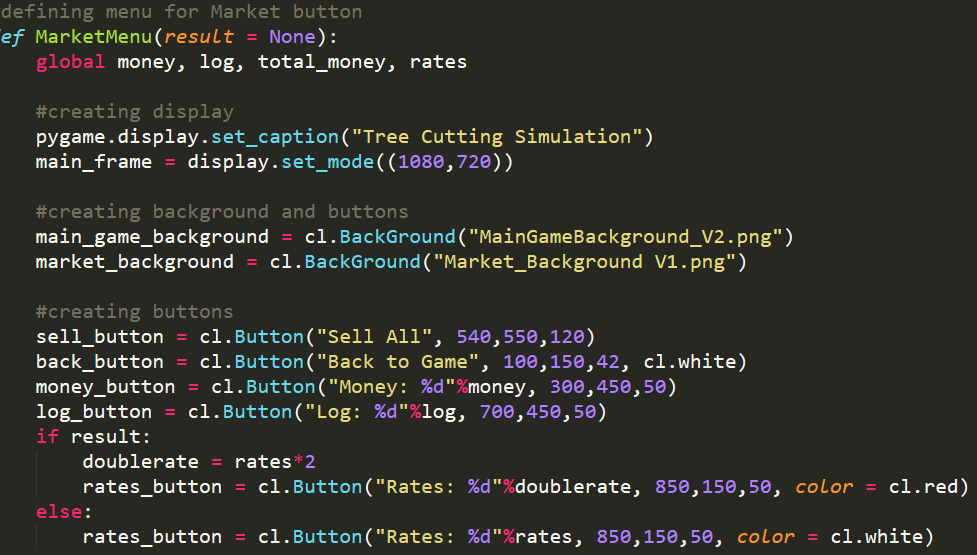


This part of the code checks if any of the variables above turns true. If move for a direction is true, the player sprite will move with respect to the direction. If swingmode is true, it will set the player into swing mode. It then will check if the player is touching a sprite from the trees group. If the player collides, it will increase the cut\_progress until the cut\_progress is greater or equal to cut\_timeneeded. Once it is true, it will remove the tree, increase log and total log by one and reset cut progress. It will also check if swingmode is false. If it is false, it will set the player back to default stance.

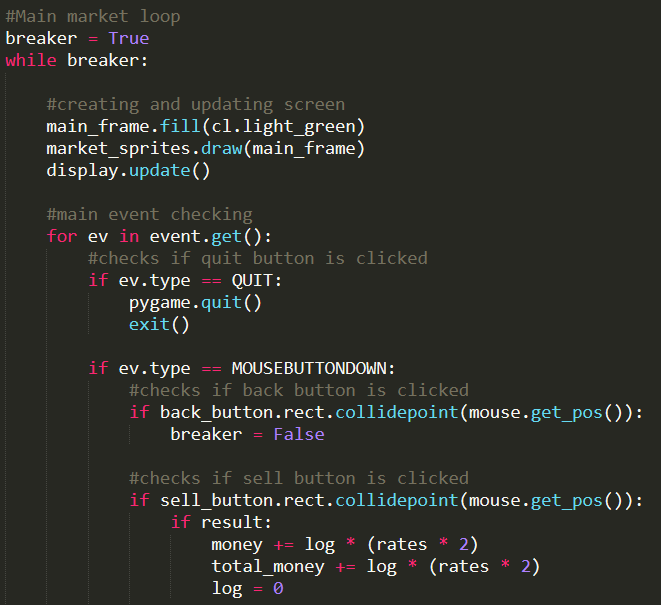


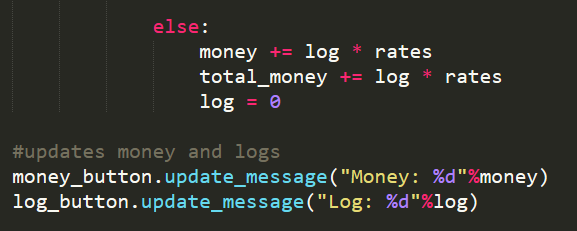
This part of the code spawns a tree every three seconds by creating a tree object and adding it to the trees group. It also checks if the mouse is hovering over sprites that are button class. If the mouse is hovering over a button sprite, it will change the button font colour into white. If the mouse is not hovering, it will change the colour back to black. At the end, it fills the screen with light green, draws trees and main game sprites and displays update.

Function MarketMenu()



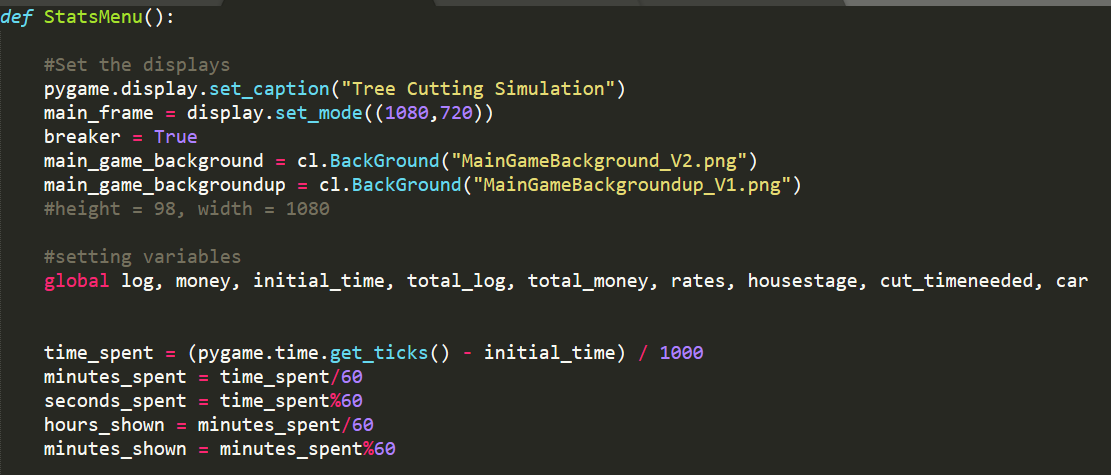
MarketMenu() is used to show the menu for the game market. The market is used as a place to convert logs into money and is the main way of earning money. The function has a parameter result with default value None. This is used to check if the player successfully won the Driving\_Phase(). If the player did, it will temporarily double the rate. If the player did not take part in Driving\_Phase(), it will use the standard rate.

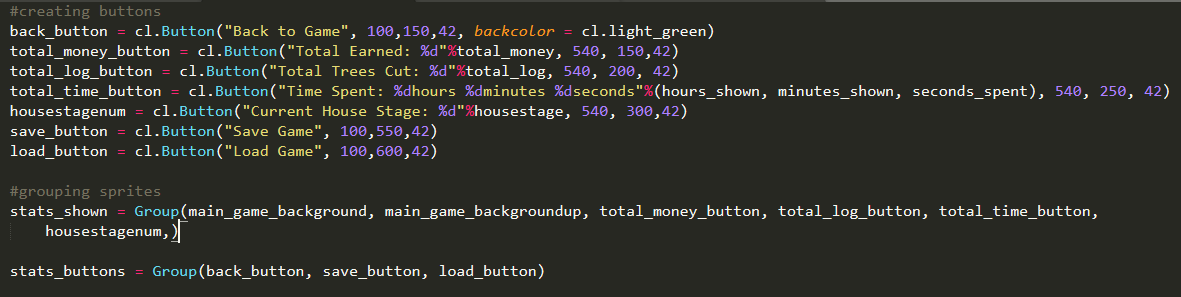




This part of the function is used to loop the display and to check events. If back button is clicked, it will break the loop. If sell\_button is clicked, it will check if result is true or not. If the result is true, rates are doubled. If not, it will use standard rates. Then, it will convert all logs into money depending on the rate.

Function StatsMenu()



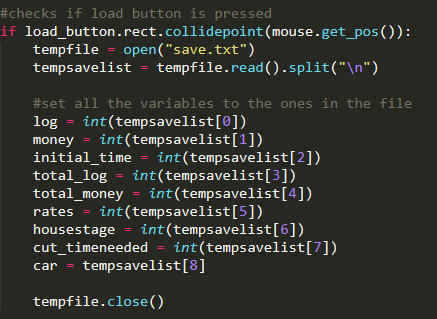


This function is used to display a screen showing statistics of the game such as total number of logs earned or time spent.

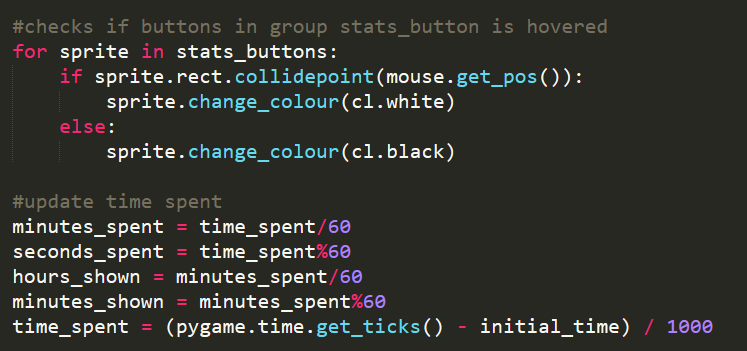
This part of the code is used to set variables such as log or time\_spent so that the buttons can show the numbers. It will also be used to save or load game progress.

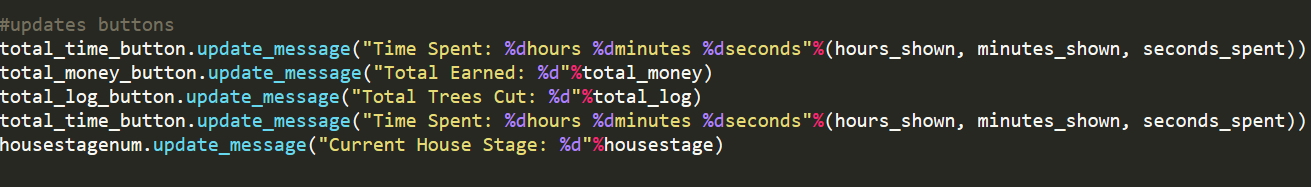


This part is used to loop the display. It is also used to check event. If event is QUIT, it will exit the code. If event is mouse click on back button, it will break the loop. If event is mouse click on save\_button, it will open a file called “Save.txt” and write down all the important variables inside it.



If event is mouse click on load\_button, it will open the “Save.txt” file and replace variables with the data saved inside the file.

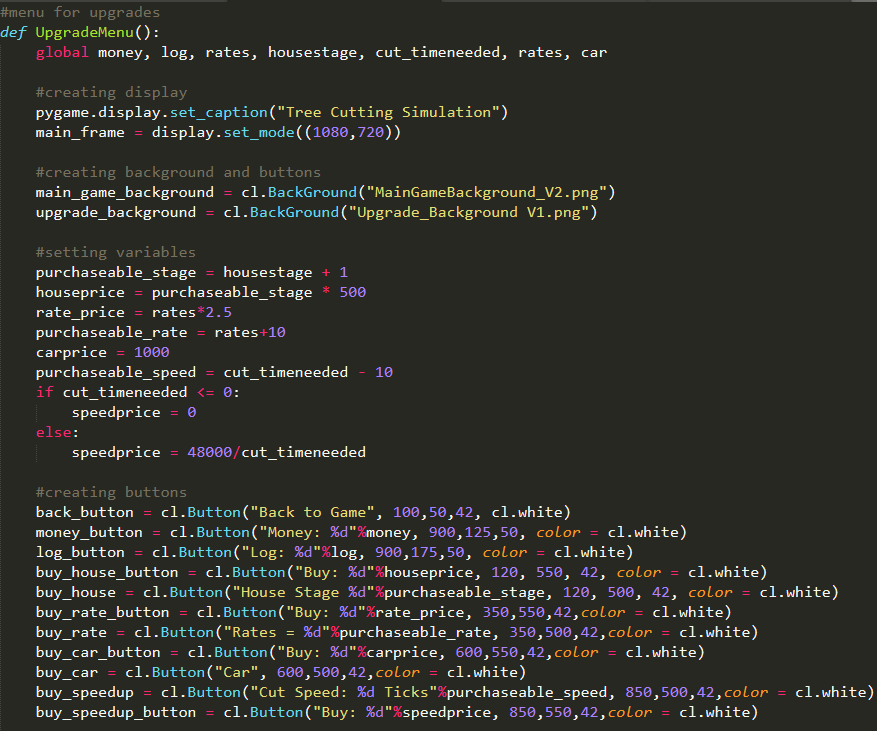


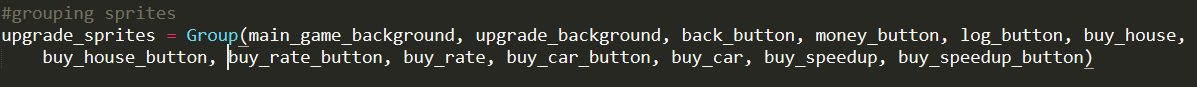


This part of the code is used to check if the mouse is hovering over the sprites in stats\_button. If it is, it will change font colour to white. If it is not, it will change font colour to black.

It is also used to update the time spent and the buttons.

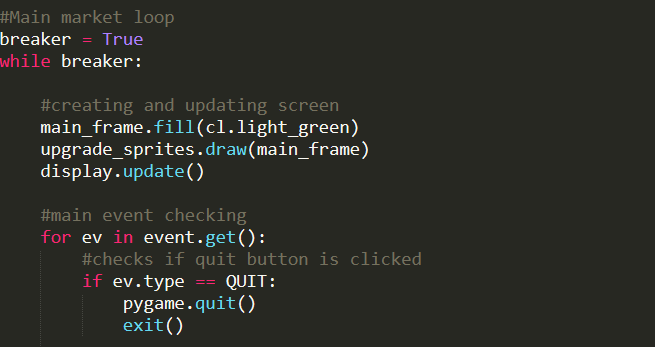
Function UpgradeMenu()





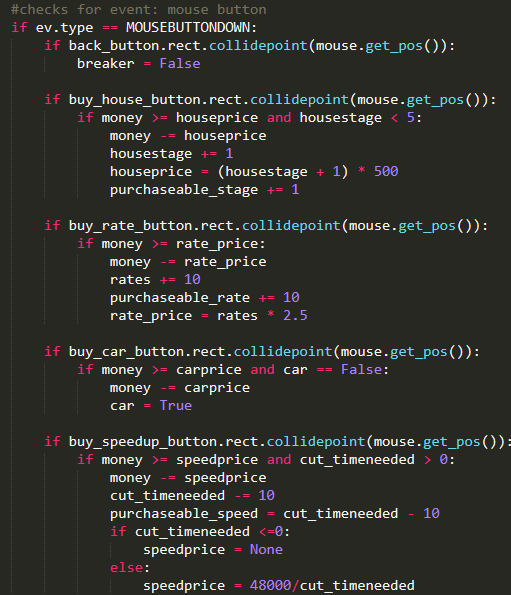
This function is used to show the upgrade menu to allow the user to buy upgrades.

This part of the function is used to set variables, buttons and sprite groups.

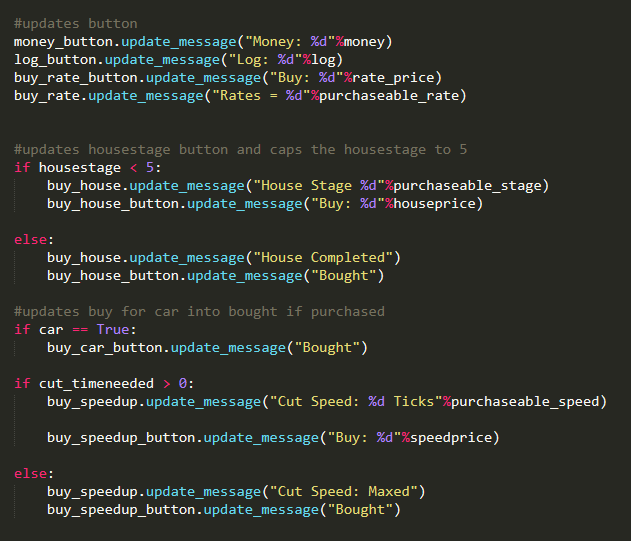


This part of the function is used to loop the display and to check events.

If the event is QUIT, it will exit.



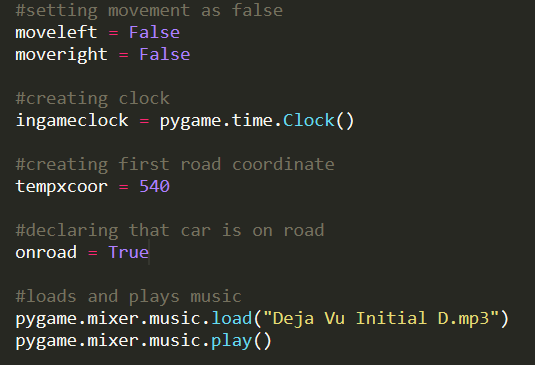
This part of the code is used to check events when the user clicks the mouse buttons. If back\_button is clicked, it will break the loop. If any of the buy buttons are clicked, it will check if the player has enough money to buy the upgrade. If there is enough money, it will decrease money with the price of the upgrade and undergo the upgrade such as increase the rate.



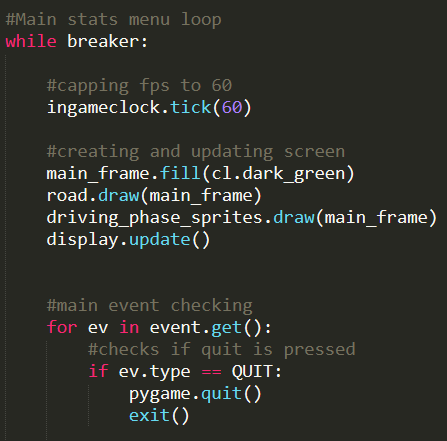
This part updates buttons and checks if any of the upgrades are already maxed. If the car variable is True, it will show bought instead of price and the player may not buy the car again. This also applies to speed upgrade and house upgrade.

Function Driving\_Phase()

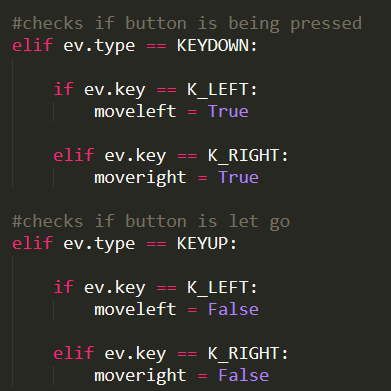




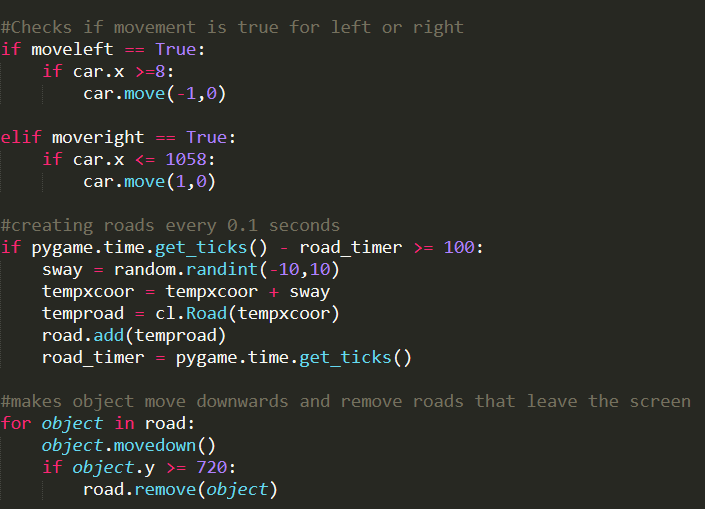
This function is used to run the driving part of the game. This part of the code is used to create variables such as onroad, objects such as car and sprite groups such as road. The mixer.music is also loaded and played here.



This part is used to loop the display, set the frame cap to 60 frame per second and to check for event. If the event is QUIT, it will exit.



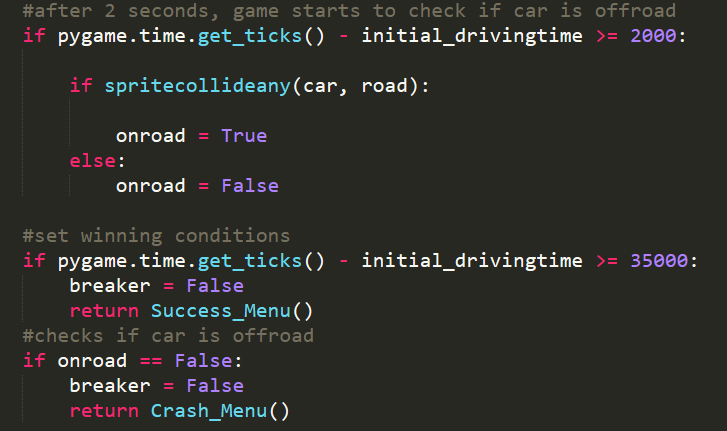
If event is KEYDOWN and left arrow or right arrow, moveleft or moveright will turn True respectively. If event is keyup for left arrow or right arrow, moveleft or moveright will turn False respectively.



This part checks if moveleft or moveright is True and will move the car left or right accordingly.

It will also check if pygame.time.get\_ticks() – road\_timer is greater or equal to 100. If it is true, it will randomize the x coordinate of the next road, add the road to the group road and resets the road\_timer.

This part also makes the objects in road to run the function movedown which is to move the road down. It also checks if the road’s y position is greater or equal to 720. If it is, it will remove the road from the group.



This part is used to check if the car goes offroad. After roughly 2 seconds, it will start checking if the car is colliding with any of the objects in the road group. If it is colliding, it will set onroad as True. If it is not on the road, it will set it as False.

If onroad is False, it will break the game and run Crash\_Menu().

If pygame.time.get\_ticks – initial\_drivingtime is greater or equal to 35000, it will break the loop and run Success\_Menu()

Function Crash\_Menu()



This part is used to show the menu when you crash in the driving\_phase. It removes 20% of log. The display loop checks for events such as clicking on the continue button. If the continue button is clicked on, the loop will break and the result False is returned.

Function Success\_Menu()

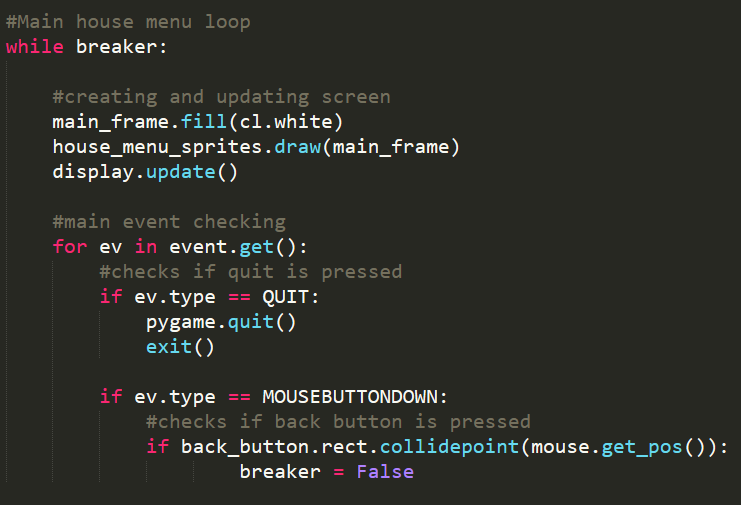


This function is used to show the menu when Driving\_Phase is completed without crashing. It is almost the same as Crash\_Menu with difference of no logs lost and returned result value is True.

Function House()



This function is used to display a picture of the player’s house. This part sets the background image depending on the variable housestage.



This part is used to loop the display and check events.

Explanation of Tree\_Cutting\_Simulation.py



This part is used to initialize pygame and to show the Start\_Menu() and Main\_Game\_Display()

Disclaimer:

The music “Déjà vu” is created by Dave Rodgers.