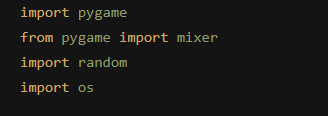
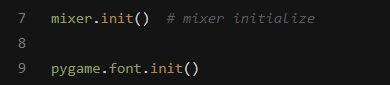
**Atari Breakout**

It is a two player game which you and your friend can enjoy.

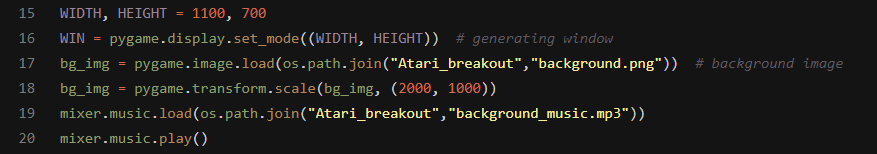
* We first started by importing the important modules required for developing this game.

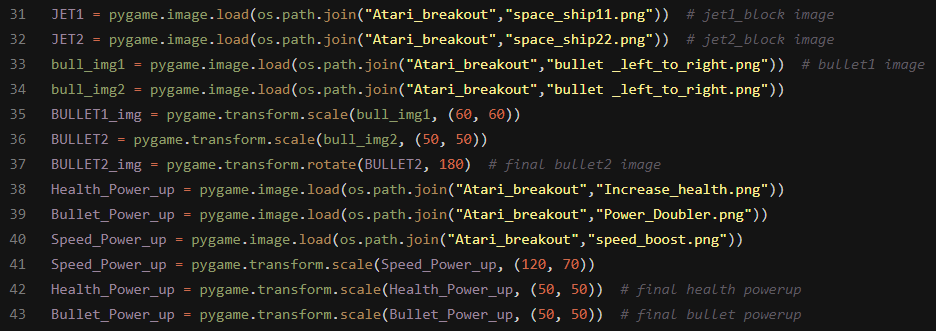
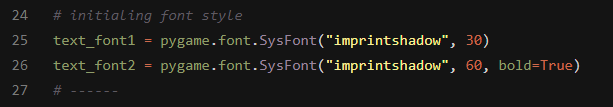
1. **Pygame** -> This module consists of computer graphics and sound that are used to design games in the python programming language.
2. **Mixer form pygame** -> In order to play music/audio files in pygame we use this.
3. **Random** -> This module is used to produce random numbers or selecting random items from the list etc.
4. **Os** -> This module provides the functions for interacting with the operating system.

* Then to start with our code we initialized our modules.



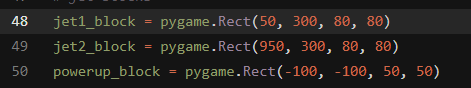
* Let's name our game **“Space Wars”** by setting the display captions using the pygame module.
* Now to create a window and for our game we started by setting the width and height of our screen by 1100 and 700 pixels.



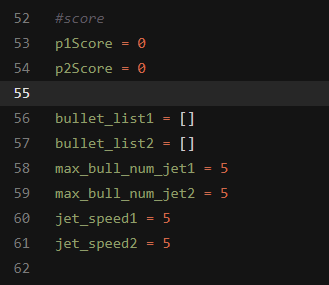
* Then we created a variable **WIN** that has our window with a given height and width.
* We imported our game background using pygame and with the help of the os module.
* We scaled the window according to our requirements.
* We imported the background music for the game using the mixer module and played it.
* Now we created some fonts that we will use later for the game.
* Here comes the most important thing the surfaces we need on the screen for our game

1. First we import 2 Jets for our player in the game,
2. We scale and adjust them using the pygame module’s several functions.
3. We import the bullet images for both the jets and transform them accordingly too.
4. We import the power-up images for our game transform them.

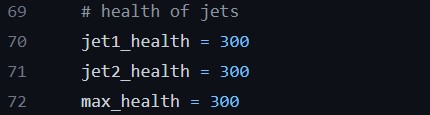
* Then we set the FPS variable to 60 and we did this so that the game functions smoothly and its functionality doesn’t differ from system to system.
* Here comes some of the variables we are going to need during the development.

1. 

These are some of the Rectangle blocks that are created using the pygame module which define the rectangle’s x-coordinate , y-coordinate , it's height and its width.

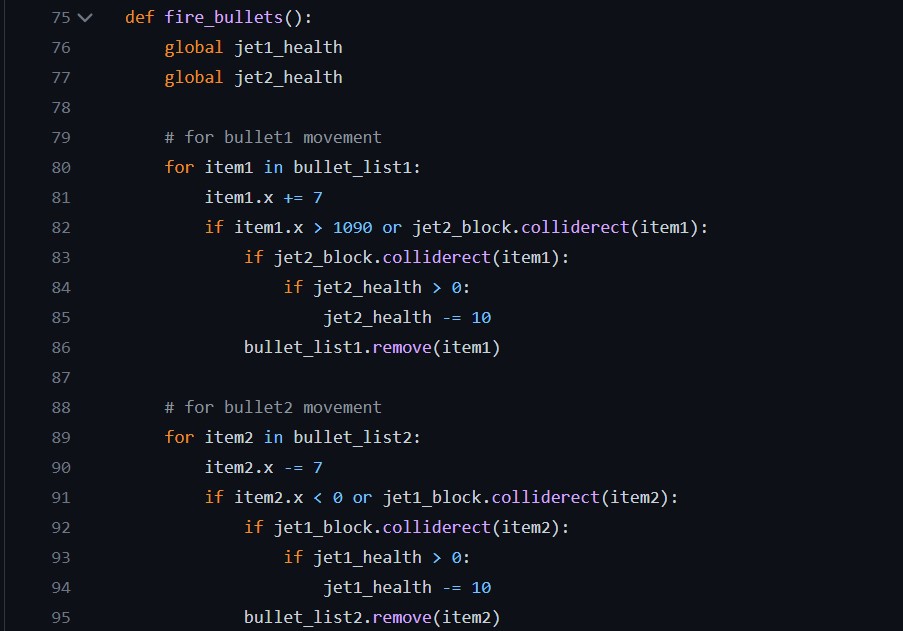
1. 

Now we have the score for each player, the list of bullets each player has, the max number of bullets each player can shoot at a time and speed for each player’s jet.



Here we set the health of each player’s jet to be 300.

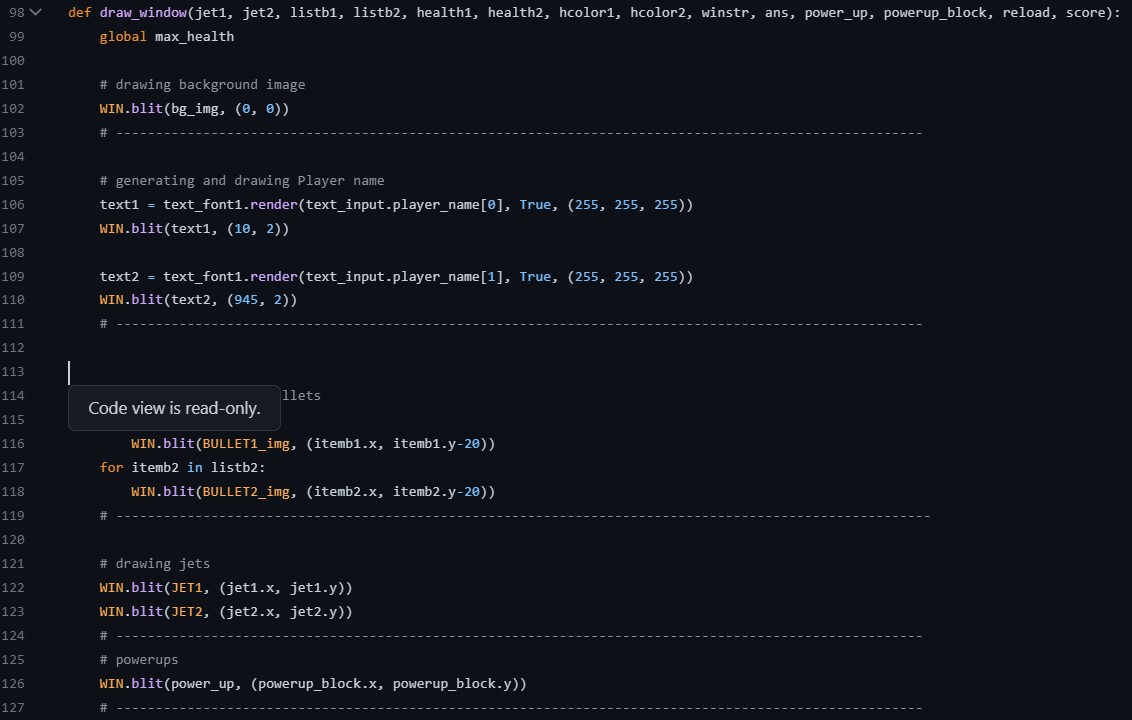
* Now time for some action with our fire bullet function.



* Here the first line presents the declaration of a function named ‘fire\_bullets’ , this function manages the movement of bullets.

1. The next two lines i.e line 73 and 74 has the health variables being declared as global.
   1. GLOBAL keyword allows the user to modify or give access to a variable outside of the current scope.
2. Next comes the “for loop” which lets the bullets move.
   1. There are two for loops with the same functionality .Both loops are present for both the space jets respectively.

* Now comes the most important function



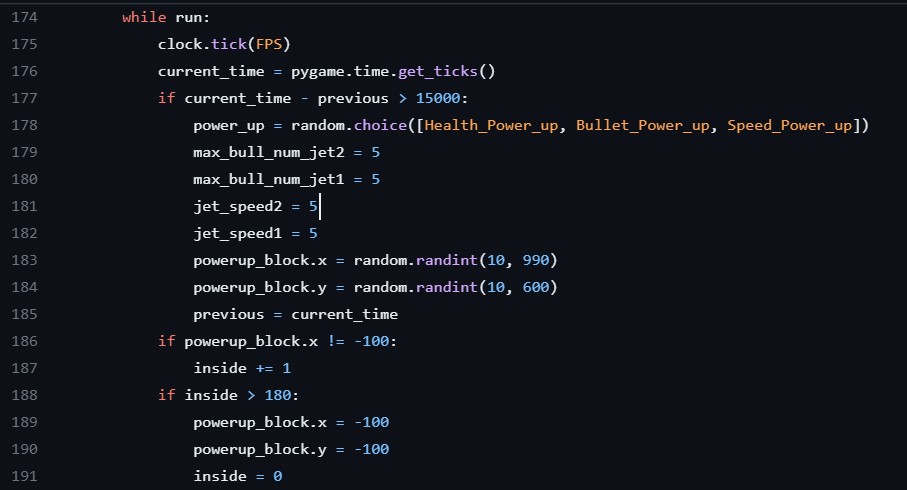
* In this fragment of code the first line , i.e. line no. 98 consists of the declaration of a function named “draw\_window()” which draws or renders every image and text on the window .
  + Firstly the background image is added .
  + Then comes rendering of the name of the players.
  + Further the jet images are put in place.
  + And lastly the power ups are added
* The above fragment consist of two pygame functions:-
  + Blit() - Blit() function renders or draws one surface over the other surface, the surface can be an image , a colored background .
  + Render() - Render() method creates a surface object i.e. an image from a text.
* The draw\_window() function is being called in the main function .
* Continuing the draw\_window function.



* This is the further part of the draw\_window function .
  + Here at top the Health Bars of the space jets are being build
  + Then further there is an “if” condition which consists of rendering of text.
  + After the “if” condition the text for the scoreboard is being rendered and displayed.
  + Lastly , the screen is updated and all the objects are displayed.
* This fragment has two more functions of pygame module:-
  + Draw.rect() - The mentioned function draws a rectangle shape on the screen to the mentioned position.
  + Display.update() - The function updates the portion of the screen, which lets the surface get displayed on the screen.
* So the last and most important and logical entity of the whole code.



* At line no. 152 the function named “main” is being declared. This function is the logical department of the code. This part of the code defines how ever object is being generated and displayed on the window.
  + Initially all the variables that are declared out of the main() function scope are declared as global.
  + Further all the **local variables** which are only being used within the main function are declared.
  + Time.Clock() - This method of the pygame module creates a new clock object that can be used to track the amount of time.
* Further section of the code:-



* The first line of this section has the “while” loop with the expression given i.e. “run”, run stores True value, this lets the loop run infinite times.
* Inside this while the “draw\_window()” function is called , so that the window is kept running until the user closes the window. As the window is closed the while loop gets terminated.
* Proceeding towards the “if ” condition , it contains the logical outlook for displaying the “power ups” on the game screen.
* The inbuilt method used in this section are :-
  + Tick() - This method when called, updates the clock by computing how many milliseconds have been passed since the previous call
  + Time.get\_ticks() - This method gets the time in milliseconds. It returns the number milliseconds since the pygame module was initialized.
  + Choice() - This is the inbuilt function of the “random” module ,this function returns the randomly selected element from a specified sequence(list , tuple or string).
  + Randint() - This is another inbuilt function of the random module. It returns an integer number selected from a specified range.
* Next section of the code:-



* This Section has the “if” condition block. This block manages collision and the effect of the collision of the jet with the power ups
* The builtin methods used is:-
  + Collide.rect() - This is the builtin method of pygame module , which tests if the two or more rectangle blocks collide or overlaps with each other.It returns boolean value i.e. is True if collision happens.