Player Class

Steven Cowie

1605240

Introduction

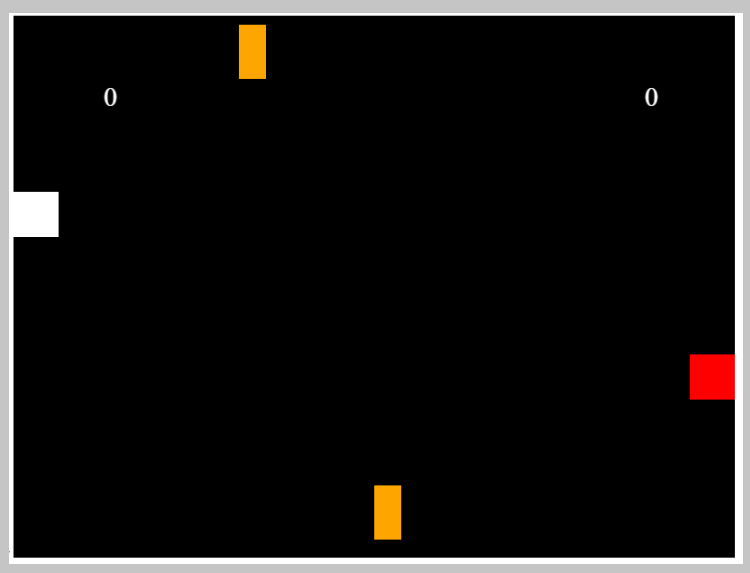
Javascript is good at providing a development platform for lightweight games. To make sure my game run as smoothly as possible I decided to make a player class in which both players would inherit almost everything they needed to have.

Below to the left is a screenshot of the game itself and what the players look like and how it plays.

Why

The ability to have 2 players moving and shooting on separate keys on the keyboard was an easy task when not put into a class system you just had to hard code it, which as we know as programmers is never the best way to do things. To clean up the code and make it more manageable I decided to make use of a player class which makes organising players much easier. As you can see directly below there is sudo code of all the code I needed for the class.

Screen shot of the game, the red and white boxes are the players, the yellow boxes are the obstacles moving up and down, you just have to shoot at each other. Scores are in the corners.



Player class Sudocode

```
Class Player{
    Spawn(X,Y,Colour){
        Player X = 0;
        Player Y = 0;
        Colour  = colour;}
Update(Up, Down){
    If (Up){
        Player Y -= playerspeed;}
    If (Down){
        Player Y += playerspeed;}}
DrawPlayer(){
    Box(X,Y, Height,Width,Colour);}
Collisions(){
    If (player x/y > canvas height/width)
    Player x/y = canvas height/width—player width;
    If (player x/y < 0)
    Player x/y = 0;
```

Solution

As mentioned before hard coding 2 players was not the right way to go about this. A player class was made where the players box was made and where it allowed me to set up the ability of the keys I wanted each player to use as well as the X & Y coordinates and speed. The player class also allowed me to be able to add in all the necessary collisions that all the players on the canvas would ever need for my game. The last thing I added was a bullet delay timer so that each of the players had use of a separate timer because before they were both sharing the same timer which led to both players not being able to shoot at the same time.

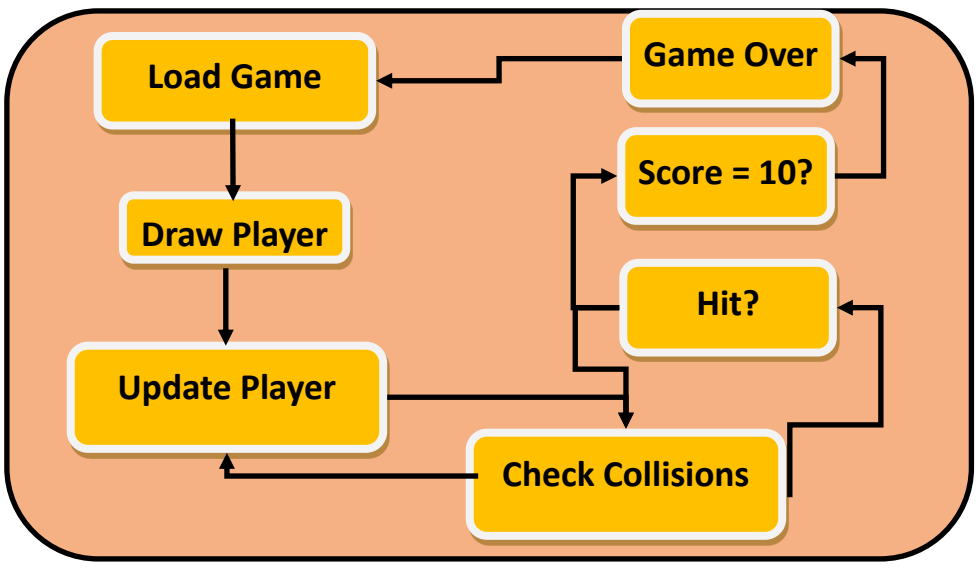
You can initialise instances of the player class in the main index file. Here you can call for it to check collisions, set up the individual players to shoot separate instances of the bullet class, set up the keys you want them to use as well as using the function to spawn in the players and the colour of the box you wanted them to be.

As seen in the bottom left is the flow of the game and how it works with the player class.

To assign individual controls you do this :

Player[0].Update(KeyW, KeyS)

Player[1].Update(KeyUpArrow, KeyDownArrow)



Conclusions

This approach worked quite well for the purpose of what I needed it to do for my simple game.

However one major room for improvement would have been to have the score as part of the player class so each individual player no matter how many were spawned would have their own score assigned to them.

To go along with this having a function built into the player class that allowed it to respawn the player at a random value on the Y axis would have been a good addition too instead of hard coding that for each player too.

Lastly getting the bullet shooting from the player class instead of again hard coding that in the game would have tidied up the code.