Topic	MARKING PLAYER & GAME CAMERA	
Class Description	**Students will use a game camera to focus the game on the active player in the game. Students will move the cars forwards with an arrow key and update the distance to the database.**	
Class	**C38**	
Class time	**45 mins**	
Goal	• Move the car forwards using arrow keys. • Update and retrieve the distance of the cars. • Add the player identifier • Introduce the concept of a game camera.	
Resources Required	• Teacher Resources ○ VS Code Editor ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen • Student Resources ○ VS Code Editor ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen	
Class structure	**WARM-UP** **Teacher-led Activity** **Student-led Activity** **WRAP-UP**	**5 mins** **15 min** **20 min** **5 mins**
WARM-UP SESSION - 5 mins		
Teacher starts slideshow  **from slides 1 to slide 9** Refer to speaker notes and follow the instructions on each slide.		

Teacher Action	Student Action
How are you doing? Are you excited to learn something new? **Run the presentation slide 1 to slide 4.** **Following are the WARM-UP session deliverables:** • Connecting the student to the previous class. • WARM-UP Quiz Session	**ESR**: Thanks, yes I am excited about it. Click on the slide show tab and present the slides.

QnA Session	
Question	**Answer**
Select the correct block of if-else, to display the correct answer in green and others in red.  A. ``` if (correctAns === allContestants[plr].answer){ fill("red") } else{ fill("green"); } ``` B. ``` if (correctAns = allContestants[plr].answer){ fill("Green") } else{ fill("red"); } ```	**D**

C.

```
if (correctAns !== allContestants[plr].answer){
  fill("Green")
}
else{
  fill("red");
}
```

D.

```
if (correctAns === allContestants[plr].answer){
  fill("Green")
}
else{
  fill("red");
}
```

Which instruction is used to give color to text?

A. fill( )
B. stroke( )
C. strokeWeight( )
D. None

A

Continue the WARM-UP session

Teacher Action	Student Action
Run the presentation from slide 4 to slide 9 and set the problem statement.	Narrate the story by using hand gestures and voice modulation methods to bring in more interest in the student.

Teacher ends slideshow

TEACHER-LED ACTIVITY - 15 mins

Teacher Initiates Screen Share

CHALLENGE

- **Create a for-in loop to display each car.**

Teacher Action	Student Action
Today, we will learn some exciting concepts about Game Camera; it allows players to follow the game characters beyond the screen. We will see how that works, but before doing that we need to move our cars.	
How should we move the car forwards using the **UP_ARROW** key?	**ESR**: Using if condition, we can change y position.
Yes. We will move the car when the game is in play mode, so let us write it in Game.js *The teacher downloads Teacher Activity 1 and opens the code in VSC.*	
Let's revise the code first. Can you explain to me the code in **sketch.js?** *Make sure to replace the database code with SDK from your own database.*	**ESR**: In **sketch.js** we use function **preload()** to add in the variables.
What is next? *The teacher can help the student to understand the code in sketch.js*	**ESR:** In function **setup()** we are creating game object and calling .**getState()** and .**start()** function from **game.js** In function **draw(),** we are checking if playerCount is 2 we update gameState to 1, and if

Fantastic! Now we have 3 objects, **form, player & game**. Our game execution code is in **game.js**. We also have the track. Can you tell me what we need next? We need to set the position of the players (cars) based on the position from the database. We will use **player.getDistance()** to get the positions of the players (cars) from the database. These functions, we will write later in the session. For now, I will create a variable index as **var index = 0**. We will use the **for-in** loop to extract the positions of the players from the **allPlayers** object. For that, we will use the **for-in** loop. This **for-in** loop has a different syntax. This is called a for-in statement; using this we look for the values inside a javascript object. Syntax: **Syntax** `for (variable in object) {` `  statement` `}` eg. for (var plr in allPlayers) { } In the first iteration, plr value will be player1 and next	gamestate is 1 we call **game.play().** **ESR:** Varied.

iteration, plr value will be player2.

The for-in loop runs for a number of elements in the object.

Here **allPlayers** variables contain the information of all the players, but we need to extract the info of each player so that we can set their position.

So we write **for(var plr in allPlayers)**.

Inside the for loop, we need to manually increase the index by one. We need to define 2 variables x and y to store the positions.

Create a for-in loop to give the position to each car inside play() in game.js.

```
play() {
  this.handleElements();
  Player.getPlayersInfo();
  if (allPlayers !== undefined) {
    image(track, 0, -height * 5, width, height * 6);

    var index = 0;
    for (var plr in allPlayers) {
      index = index + 1;
      var x = allPlayers[plr].positionX;
      var y = height - allPlayers[plr].positionY;

      cars[index-1].position.x= x;
      cars[index-1].position.y = y;
    }
```

To access the data from the **allPlayers** object we write the name of the **object[plr].positionX**. This value will go in **var x**.

<table>
<tr>
<td>The same process we will do for the y-direction but we need to subtract the y position from the height because we want to place the player at the bottom of the screen.

Then we need to set these positions to the position of the car's sprite as well. ‘Cars’ is an array containing 2 car sprites; we need to set the position to both of them.

The index value is 1 now, but we need to keep it 0 because the first element of the array is at 0. That's why we need to subtract 1 from the index.

Hence using cars[index-1] we assign x and y positions to each car.

The next step is to make the player move by keypress, which we are implementing by using the keyIsDown() function of p5.js.

To make sure our code is structured we will create a separate function to add controls to the car.
let us name it as handlePlayerControls().

Can you tell me where do we call this function?

Yes, and because it is defined in the same file Game.js we can call it as this.handlePlayerControls().</td>
<td>ESR: In Play().</td>
</tr>
<tr>
<td colspan="2">Call this.handlePlayerControls() after for-in loop of play()</td>
</tr>
</table>

```
for (var plr in allPlayers) {
    //add 1 to the index for every loop
    index = index + 1;

    //use data form the database to display the cars in x and
    var x = allPlayers[plr].positionX;
    var y = height - allPlayers[plr].positionY;

    cars[index - 1].position.x = x;
    cars[index - 1].position.y = y;
}

this.handlePlayerControls();

drawSprites();
```

Create a condition to move the car upwards inside handlePlayerControl() of game.js.

```
handlePlayerControls() {
// handling keyboard events
    if (keyIsDown(UP_ARROW)) {
        player.positionY += 10;
        player.update();
    }
}
```

The condition checks for the keypress event and once the key is pressed we are changing the y position of the player and we are calling the **player.update()** function so that player's position will be updated in the database.

We have called the update method here but we need to write this method in the class.

First, create the **update()** method; in this, we are using

playerIndex to update the **x and y position** of the player.	

Create update() in player.js.

```
update() {
  var playerIndex = "players/player" + this.index;
  database.ref(playerIndex).update({
    positionX: this.positionX,
    positionY: this.positionY,
  });
}
```

In the update() method, we are referring to player1 & player2 to update their respective position on the canvas. Can you explain this method? Let us run the code and check if the cars are moving. *The teacher runs the code using GoLive.Remember to reset the database by:* *gameState= 0* *playerCount =0* *Delete Players*	**ESR:** We use **.ref()** to refer to location in the database. **.update()** is used to update an existing field in the database with a new value. Here we are updating the x & y positions of the car.

OUTPUT:

What should we do next? We need to also retrieve the distance from the database so that the other player knows where another car is. Secondly, how are you able to identify which one is my car? For this, we will create a small circle below the car. With this guide in mind, why don't you start with the coding exercise? I will be guiding you in the exercise. Now it's your turn. Please share your screen with me.	ESR: Varied ESR: Varied

Teacher Stops Screen Share

STUDENT-LED ACTIVITY - 20 mins

- **Ask the student to press the ESC key to come back to the panel.**
- **Guide the student to start Screen Share.**
- **The teacher gets into Fullscreen.**

ACTIVITY

- **Mark active player**
- **Game Camera**
- **create getDistance() method in player.js**

Teacher starts slideshow  **for slide 10 and slide 14**
Refer to speaker notes and follow the instructions on each slide.

Teacher Action	Student Action
The teacher guides the student to open Student Activity 1. Download the zip folder, unzip it and save it as C38. *Make sure to add the database information in **index.html.*** *During the class, if a student is changing any position of any element or object in the code/ ask them to write it somewhere to use in codes of the next classes.* *In case any x or y positions were changed by the student in the code in the previous class, make those changes to fit into his screen size.*	*Alternatively, the student can clone **Student Activity 1.***

Let's create the getDistance method inside the **player.js** file.

Here first we are getting the position values from the database using **database.ref()**. Then we are storing the data in the data variable. Data coming from the database is in the form of a JavaScript object so you need to extract the x and y position of the car.

Create 2 variables this.positionX and this.positionY and we assign these to the x and y position values from the database.

Create getDistance() in player.js

```
getDistance() {
   var playerDistanceRef = database.ref("players/player" + this.index);
   playerDistanceRef.on("value", data => {
     var data = data.val();
     this.positionX = data.positionX;
     this.positionY = data.positionY;
   });
 }
```

Here we are using properties created in the **constructor()** to update the distance.

So where will you call this function?

Yes, we can do that, however, in order to make sure cars are always placed at the right position when the game starts, we will add it in the **handlemousePressed()** function of **Form.js.**

ESR: In the **play()** of game.js.

The teacher can recall concepts of what is covered in handleMousePressed().	
Call player.getDistance() in Form.js  	
Now, let's draw a small circle below the playerCar to highlight the active player in a window Where should we draw the circle? Why?	**ESR**: Inside **play()** in Game class as this is the function which is called when the game is in the 'play' state.
Where are we identifying the current player in the game? We identify the current player by matching **index** with **player.index** inside the condition - **if(index===player.index).**	**ESR**: Varied/using index.

Let's give instructions to add the circle at the x and y position of the car using **ellipse()** when the condition is **true.** Perfect, you can also use **fill()** to make an identifier red in color and **stroke()** to give its border width.	*The student uses the **ellipse()** function to add the circle.* *The student can also use **stroke()** and **fill()** to give color to the circle.*
Inside play() method of game.js `if (index === player.index) {` `  stroke(10);` `  fill("red");` `  ellipse(x, y, 60, 60);`	
Now, let us run the code to check. What issue are we facing currently? When we keep pressing the up arrow till the car goes out of the screen and we are not able to see the cars. All the players see the same thing in the game. This is where the concept of a **Game Camera** comes in. **The game camera** allows us to change how and from where we are viewing the game. We want the **Game Camera** to be focused on each player's car. We can set the camera position in the game differently for each player. *Guide the student to set the camera position for each player in the game.*	*The student runs the code; to see the car move with a highlighted circle drawn just below it.* ESR: Varied.

Here we are setting the camera position x and position y according to the player's car position. We will restrict the movement of cars to stay on track. For that, we can keep the x position of the camera to align with the x position of a car. Our cars move forward, so we will align the position of the camera with the position of the car.	*The student sets the camera position for each player in the game.*

Adding Camera on each player's car in play()

```
if (allPlayers !== undefined) {
  image(track, 0, -height * 5, width, height * 6);

  //index of the array
  var index = 0;
  for (var plr in allPlayers) {
    //add 1 to the index for every loop
    index = index + 1;

    //use data form the database to display the cars in x and y direction
    var x = allPlayers[plr].positionX;
    var y = height - allPlayers[plr].positionY;

    cars[index - 1].position.x = x;
    cars[index - 1].position.y = y;

    if (index === player.index) {
      stroke(10);
      fill("red");
      ellipse(x, y, 60, 60);

      // Changing camera position in y direction
      camera.position.x = cars[index - 1].position.x
      camera.position.y = cars[index - 1].position.y;
    }
  }
```

Let's reset and run the code. Make sure gameState and playCount is 0 and delete all players. *(Make sure gameState and playCount is 0 and delete all players.)* Wow! Great! Now the player will know which car is his/hers. And also the Game Camera allows us to see the car beyond the canvas. Good work!	*The student runs the code on the browser by logging in for each user to check the output.* *The student presses the up arrow key in different tabs to see the car sprites move and different camera angles for each player's car.*

OUTPUT:

Teacher Guides Student to Stop Screen Share	
WRAP-UP SESSION - 5 Mins	
Teacher starts slideshow  **from slides 15 to slide 25.** Refer to speaker notes and follow the instructions on each slide.	
Teacher Action	**Student Action**
Run the presentation from slide 15 to slide 25. **Following are the WRAP-UP session deliverables:** • **Revise the concepts** • **Wrap Up Quiz** • **Explain the facts and trivia** • **Project for the day** • **Next class challenge**	Guide the student to develop the project and share it with us
Quiz time - Click on in-class quiz	
Question	**Answer**
What does the **handlePlayerControls()** function of p5.play do? A. to control the movement of car B. to control the gameState C. to control the playerCount D. to control the database connection	**A**
What is this piece of code being written for? if (index=== player.index){ ellipse(x, y, 60, 60); } A. to mark the current car by drawing a circle below it B. to match the index with player.index C. to give a different color to all players except the active player in the browser	**A**

D. to track the path followed by each car	
What is the use of a Game Camera? A. Game camera allows us to change how and from where we are viewing the game. B. We can focus the game camera on each player's car. C. We can set the camera position in the game differently for each player. D. All of the above	**D**
End the quiz panel	
FEEDBACK • **Encourage the student to make reflection notes in markdown format.** • **Compliment the student for her/his effort in the class.** • **Review the content of the lesson.**	
Teacher Action	**Student Action**
You get Hats off for your excellent work! In the next class, you will create a leader board and reset button to reset the values in the database.	*Make sure you have given at least 2 Hats Off during the class for:* 
*** This Project will take only 30 mins to complete. Motivate students to try and finish it immediately after the class.*** **Project Overview**	*Student engage with the teacher over the project.*

KANGAROO IN JUNGLE-1

Goal

In class 38, you learned how to play around with the camera. In this project, you have to practice and apply what you have learned in the class and create a game where a kangaroo searches for food. Add a camera in the game moving in the x-direction.

Story:

Joey is a baby kangaroo who is hungry and is eating all the tasty shrubs and plants in the jungle. Help Joey find and eat as many shrubs as he can. Add obstacles and a scoring system to make the game interesting.

I am very excited to see your project solution and I know you will do really well.

Bye Bye!

Teacher ends slideshow 

Teacher Clicks ✖ End Class

ADDITIONAL ACTIVITIES

Encourage the student to write reflection notes in their reflection journal using markdown.

Use these as guiding questions:

- What happened today?

Student uses the markdown editor to write her/his reflection as a reflection journal.

○ Describe what happened ○ Code I wrote ● How did I feel after the class? ● What have I learned about programming and developing games? ● What aspects of the class helped me? What did I find difficult?	

Activity	Activity Name	Links
Teacher Activity 1	p5 BoilerPlate for Multiplayer Car Racing Game	https://github.com/pro-whitehatjr/C38RV_SpeedRacer_TeacherActivity
Teacher Activity 2	Teacher Reference Code	https://github.com/pro-whitehatjr/C38RV_SpeedRacer_ReferenceCode
Student Activity 1	p5 BoilerPlate for Multiplayer Car Racing Game	https://github.com/pro-whitehatjr/C38RV_SpeedRacer_StudentActivity
Teacher Reference	Visual aid link	https://curriculum.whitehatjr.com/Visual+Project+Asset/PRO_VD/PRO_C38_V3_Iit_WithCues.html
Teacher Reference	In-class quiz	https://s3-whjr-curriculum-uploads.whjr.online/9c7e3c55-be72-4b57-b760-efbcb7bf90aa.pdf
Project Solution	Kangaroo in Jungle-1	https://github.com/pro-whitehatjr/Project-C38-V3