



Tema	Juego en la Base de Datos	
Descripción de la Clase	Los alumnos crean el juego de carreras de autos multijugador, que ocurre solo en la base de datos. Los alumnos presionan la tecla de flecha hacia arriba, para cambiar la propiedad de distancia recorrida de cada jugador, en la base de datos. Las distancias recorridas por todos los jugadores, se muestran en la pantalla con el jugador que juega el juego resaltado.	
Clase	C37	
Duración de la Clase	45 minutos	
Objetivo	<ul style="list-style-type: none"> • Estructurar el código del juego para incluir más propiedades, y funciones para cada clase. • Escribir el código de comportamiento, en el estado de juego del juego, cuando todos los jugadores están conectados. • Mostrar a todos los jugadores con sus puntuaciones de distancia recorrida. 	
Recursos Requeridos	<ul style="list-style-type: none"> • Recursos para Profesores <ul style="list-style-type: none"> ○ Laptop con conectividad a internet ○ Auriculares con micrófono ○ Cuaderno y bolígrafo • Recursos para Alumnos <ul style="list-style-type: none"> ○ Laptop con conectividad a internet ○ Auriculares con micrófono ○ Cuaderno y bolígrafo 	
Estructura de la Clase	Rompiendo el Hielo Actividad dirigida por el Profesor Actividad dirigida por el Alumno Conclusión	5 minutos 10 minutos 25 minutos 5 minutos
<p style="text-align: center;"><u>CONTEXTO</u></p> <ul style="list-style-type: none"> • Cómo estructurar el código, facilita la escritura y la depuración del código. 		
Pasos de Clase	Acción del Profesor	Acción del Alumno

Paso 1: Rompiendo el Hielo (5 minutos)	¡Hola! ¿Estás emocionado de seguir construyendo el juego de carreras de autos?	REA: ¡Sí!
	Primero, recordemos en dónde nos quedamos en la última clase.	REA: Creamos un formulario que permite a los jugadores iniciar sesión y ver el saludo. El nombre del jugador se registra en la base de datos.
	<p>¡Tengo una emocionante pregunta de prueba para ti! ¿Estás listo para responder a esta pregunta?</p> <p><i>Por favor haga clic en el botón</i></p>  <p>que está en la esquina inferior derecha de su pantalla para iniciar el Cuestionario en Clase.</p> <p><i>El cuestionario será visible para usted y el alumno.</i></p> <p><i>Anime al alumno a responder la pregunta del cuestionario.</i></p> <p><i>Puede que el alumno elija la opción incorrecta. Ayúdele a pensar bien la pregunta y pídale que responda de nuevo.</i></p> <p><i>Cuando elija la opción correcta el</i></p>  <p><i>botón aparecerá en su pantalla.</i></p>	REA: Sí.

	<i>Haga clic en Finalizar Cuestionario para cerrar la ventana y continúe con la clase.</i>	
	<p>¡Estupendo! También habíamos hablado de la importancia de estructurar nuestro código. Estructurar el código, nos ayuda a escribir un mejor código.</p> <p>Es muy parecido a organizar tu habitación. Si tu habitación está desorganizada, no podrás encontrar dónde has guardado tus cosas. También, será difícil decidir dónde guardar las cosas nuevas en tu habitación.</p> <p>De manera similar, un código mal organizado, dificultaría la búsqueda de errores en tu código. También sería difícil escribir el código nuevo, en la parte superior de tu código escrito previamente, sin causar errores.</p> <p>Se necesita un poco de tiempo para pensar en cómo estructurar tu código; pero todo el tiempo dedicado a hacer que tu código se vea bien, estructurado y más legible, ¡siempre vale la pena!</p>	<i>El alumno escucha, comprende y hace preguntas.</i>
	<p>¡En la clase de hoy, crearemos un juego de carreras de autos que se ejecuta solo en la base de datos!</p> <p>Comencemos.</p>	-

El Profesor Inicia Compartir Pantalla

DESAFÍO

- Vuelve a estructurar el código para agregar más propiedades y funciones deseadas en el código.
- Crea el juego de carreras de autos en la base de datos.

Paso 2: Actividad dirigida por el Profesor (10 minutos)

Permítanme mostrarles rápidamente lo que estaremos construyendo el día de hoy.

El profesor clona el código del enlace de referencia, agrega la configuración de firebase para su propia base de datos, y hace los cambios necesarios en la estructura de la base de datos de firebase.

Asegúrate de que gameState y playerCount sean 0.
¿Sabes por qué es importante?

El profesor ejecuta el código en el servidor web, abre 4 pestañas diferentes, e ingresa los nombres de los jugadores para mostrar el inicio del juego.

El profesor presiona la tecla de flecha hacia arriba en los diferentes navegadores, para mostrar el número que se muestra en los diferentes jugadores.

El profesor también muestra las entradas en la base de datos.

El alumno observa.

REA:

Esos son los estados iniciales del juego.

The image shows two screenshots related to a Firebase project. The top screenshot is from the Firebase console's 'Database' tab, showing the 'Realtime Database' for the project 'multiplayer-car-racing-game'. A warning message states: 'Your security rules are defined as public, so anyone can steal, modify, or delete data in your database'. The database structure is shown as follows:

```
multiplayer-car-racing-game
├── gameState: 0
└── playerCount: 0
```

The bottom screenshot is from a code editor (VS Code) showing the 'index.html' file. It contains the following code:

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.9.0/addons/p5.dom.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.9.0/addons/p5.sound.min.js"></script>
<script src="p5.play.js"></script>

<script src="https://www.gstatic.com/firebasejs/6.3.4/firebase-app.js"></script>
<script src="https://www.gstatic.com/firebasejs/6.3.0/firebase-database.js"></script>

<script>
  //Your web app's Firebase configuration
  // Add your firebase config here

  firebase.initializeApp(firebaseConfig);
</script>
```

A large yellow box with the text 'Add your firebase config here' is overlaid on the code. A notification at the bottom right says 'There is an available update.' with buttons for 'Download Now', 'Later', and 'Release'.


Después de que algún jugador inicie sesión:

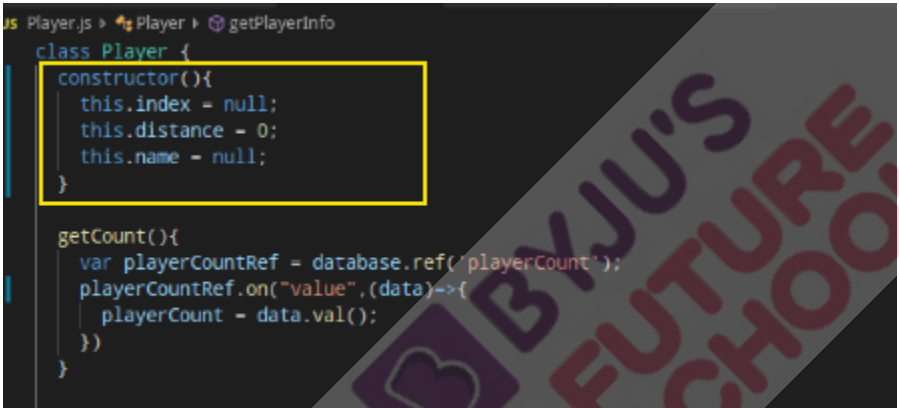


Después de que los 4 jugadores inicien sesión:



	¿Qué está sucediendo aquí?	<p>REA: Tan pronto como 4 jugadores inician sesión, el formulario y el saludo desaparecen.</p> <p>Cuando comienza el Juego, todos los jugadores y la distancia recorrida por ellos se muestran en la pantalla.</p> <p>Se resalta el jugador que inició sesión. La distancia actual recorrida por el jugador, aumenta cuando se presiona la tecla de flecha hacia arriba.</p>
	<p>¿Qué crees que podría estar sucediendo en el código?</p> <p>¿Cómo crees que conseguiremos el resultado que acabas de ver?</p>	<p>REA: variado</p>
	Codifiquemos para esto. Mientras codificamos, siempre intentaremos mantener nuestro código estructurado.	-
El Profesor Detiene Compartir Pantalla		
	Ahora es tu turno. Comparte tu pantalla conmigo.	
<ul style="list-style-type: none"> • Pídale al Alumno que presione la tecla ESC para volver al panel • Guíe al Alumno para que comience a Compartir Pantalla • El Profesor entra en Pantalla Completa 		

ACTIVIDAD <ul style="list-style-type: none"> • Escribe el código de comportamiento en el estado de juego del juego, cuando todos los jugadores están conectados. • Muestra a todos los jugadores con sus puntuaciones de distancia recorrida. 		
Paso 3: Actividad dirigida por el Alumnos (25 minutos)	<p>Guíe al alumno para que abra el proyecto de la última clase en VS Code, y habilite el uso compartido en vivo.</p> <p>El profesor abre el enlace e inicia la colaboración en vivo.</p>	<p>El alumno abre el proyecto de la última clase o Clona el proyecto de la Actividad del Alumno 1.</p> <p>El alumno habilita la función de compartir en vivo, y comparte el enlace con el profesor.</p>
	<p>Guíe al alumno para que cambie firebaseConfig en index.html.</p>	<p>El alumno cambia firebaseConfig en index.html.</p>
	<p>Guíe al alumno para que cambie la estructura de la base de datos de firebase como se muestra en la siguiente imagen.</p>	<p>El alumno cambia la estructura de la base de datos de firebase como se describe en la imagen a continuación.</p>
		
	<p>Comencemos con la Clase Player (Jugador).</p> <p>Agreguemos nuevas propiedades y funciones a nuestra clase Player,</p>	

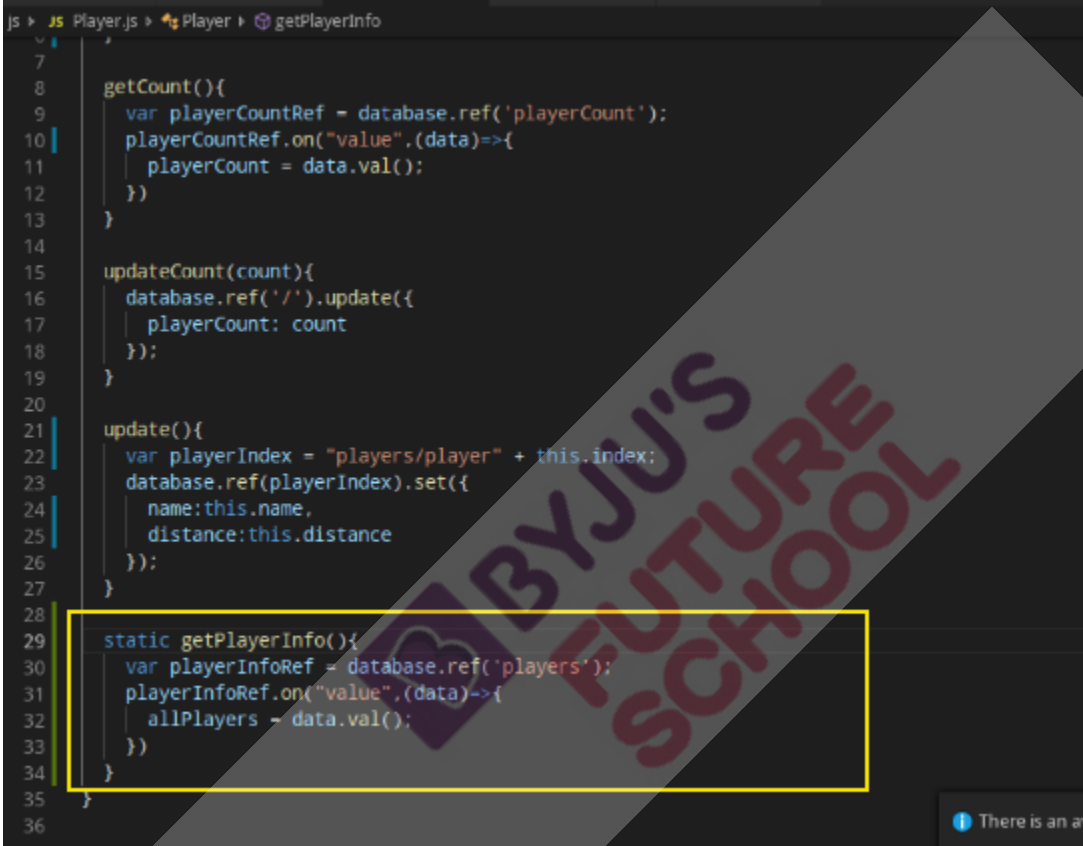
	<p>para satisfacer las necesidades de nuestro nuevo programa.</p> <p>El constructor en nuestra clase Player está vacío. ¿Cuáles son las propiedades de un objeto de un jugador?</p>	<p>REA: nombre, distancia recorrida, índice del jugador.</p>
	<p>Exactamente, creemos estas propiedades dentro del constructor del jugador.</p>	<p><i>El alumno escribe el código como se muestra en la imagen a continuación.</i></p>
 <pre> js Player.js > Player ▶ getPlayerInfo class Player { constructor(){ this.index = null; this.distance = 0; this.name = null; } getCount(){ var playerCountRef = database.ref('playerCount'); playerCountRef.on("value",(data)->{ playerCount = data.val(); }) } } </pre>		
	<p>Las funciones getCount() y updateCount() seguirán siendo las mismas.</p> <p>Usemos la función update() para actualizar tanto el nombre, como la distancia del jugador.</p> <p>Además, necesitamos cambiar la referencia a la base de datos, ya que todos los jugadores ahora estarán dentro de "jugadores" en la base de datos.</p> <p>Usaremos la propiedad 'index - índice' del jugador, para actualizar los</p>	<p><i>El alumno, con la guía del profesor, escribe la función update() como se muestra en la imagen a continuación.</i></p>

valores de ese jugador en particular en la base de datos.

```
JS ▶ JS Player.js ▶ Player ▶ getPlayerInfo
7
8   getCount(){
9     var playerCountRef = database.ref('playerCount');
10    playerCountRef.on("value",(data)=>{
11      playerCount = data.val();
12    })
13  }
14
15  updateCount(count){
16    database.ref('/').update({
17      playerCount: count
18    });
19  }
20
21  update(){
22    var playerIndex = "players/player" + this.index;
23    database.ref(playerIndex).set({
24      name:this.name,
25      distance:this.distance
26    });
27  }
28
```

```
JS sketch.js ▶ draw
1  var canvas, backgroundImage;
2
3  var gameState = 0;
4  var playerCount;
5  var allPlayers;
6  var distance = 0;
7  var database;
8
9  var form, player, game;
10
11
12  function setup(){
13    canvas = createCanvas(400,400);
14    database = firebase.database();
15    game = new Game();
16    game.getState();
17    game.start();
18  }
19
```

	<p>Escribamos una función para obtener toda la información de los jugadores.</p> <p>Esta función no se adjuntará a ningún jugador en particular. Podemos declararlo como una función estática. Las funciones estáticas no se adjuntan a cada objeto de la clase. Estamos tratando de obtener toda la información de los jugadores aquí; el trabajo no involucra ningún objeto en particular.</p> <p>Las funciones estáticas son aquellas funciones comunes, que son llamadas por la propia clase en lugar de por los objetos de la clase. Usamos la palabra clave 'estática' antes de la función, para convertirla en una función estática. Pronto aprenderemos a usarla.</p> <p>Primero, declaremos una función estática que obtiene todos los datos de los jugadores y los almacena.</p> <p>Los datos de los jugadores se almacenarán como JSON, ya que la estructura de la base de datos firebase es de tipo JSON.</p> <p>¿Recuerdas la estructura de datos JSON? ¿Dónde se usa?</p>	<p><i>El alumno escribe la función estática <code>getPlayerInfo()</code> para capturar datos de todos los jugadores en una variable llamada "allPlayers" (todos los Jugadores).</i></p> <p><i>El alumno también declara una variable global llamada <code>allPlayers</code>.</i></p> <p>REA: SÍ { key1: item1, key2: item2, </p>
--	---	--

		<pre>}</pre> <p>Se utiliza para almacenar datos de forma organizada.</p>
	 <pre> 7 8 getCount(){ 9 var playerCountRef = database.ref('playerCount'); 10 playerCountRef.on("value",(data)=>{ 11 playerCount = data.val(); 12 }) 13 } 14 15 updateCount(count){ 16 database.ref('/').update({ 17 playerCount: count 18 }); 19 } 20 21 update(){ 22 var playerIndex = "players/player" + this.index; 23 database.ref(playerIndex).set({ 24 name:this.name, 25 distance:this.distance 26 }); 27 } 28 29 static getPlayerInfo(){ 30 var playerInfoRef = database.ref('players'); 31 playerInfoRef.on("value",(data)->{ 32 allPlayers = data.val(); 33 }) 34 } 35 36 </pre>	
	<p>Abramos la Clase Form (Formulario).</p> <p>Podemos definir todas las propiedades del formulario: entrada, botón, elemento de saludo dentro del constructor de la clase.</p> <p>La entrada, botón y saludo dentro de la función mousePressed() debe cambiarse a this.input, this.button y this.greeting.</p>	<p><i>El alumno escribe el código dentro de la clase Form, como se muestra en la imagen a continuación.</i></p>

	<p>Además, "this – esto" se refiere al objeto que llama a la función. Queremos que 'this' dentro de la función mousePressed, se refiera al objeto formulario. Sin embargo, el elemento de botón está llamando a la función mousePressed() y 'this' aquí, se refiere al elemento de botón. Esto no es lo que queremos. Para que 'this' continúe refiriéndose al objeto formulario, usamos funciones de flecha.</p> <p>Las funciones de flecha unen la función al objeto original que la llama.</p> <p>Aquí se llama mousePressed dentro de la función display (visualización), que es llamada por el objeto formulario. ()=> La función de flecha asegura que 'this' permanece vinculado al objeto formulario.</p> <p><i>Guíe al alumno para que haga cambios en el código, y use una función de flecha.</i></p> <p><i>También guíe al alumno a escribir una función hide() que será llamada cuando cambie el estado del juego, y cuando queramos ocultar el formulario.</i></p>	
--	--	--

```

js ▶ JS Form.js ▶ Form ▶ hide
1  class Form {
2
3      constructor() {
4          this.input = createInput("Name");
5          this.button = createButton('Play');
6          this.greeting = createElement('h2');
7      }
8      hide(){
9          this.greeting.hide();
10         this.button.hide();
11         this.input.hide();
12     }
13
14     display(){
15         var title = createElement('h2')
16         title.html("Car Racing Game");
17         title.position(130, 0);
18
19         this.input.position(130, 160);
20         this.button.position(250, 200);
21
22         this.button.mousePressed(()->{
23             this.input.hide();
24             this.button.hide();
25             player.name = this.input.value();
26             playerCount+=1;
27             player.index = playerCount;
28             player.update();
29             player.updateCount(playerCount);
30             this.greeting.html("Hello " + player.name)
31             this.greeting.position(130, 100);
32         });
33
34

```

Pasemos a la Clase Game (Juego).

Escribamos una función play() que se llamará cuando el estado del juego alcance el estado play o 1. Dentro de la función play(), oculta el formulario.

Obtén todos los datos de los jugadores, y muéstralos en la pantalla.

Cambia la distancia, y actualízala en la base de datos cuando se presione la tecla de flecha 'ARRIBA'.

El alumno escribe el código como se muestra en la imagen a continuación.

Nota: usamos 'plr' porque el jugador ya está definido.

```

6   form = new Form()
7   form.display();
8   }
9   }
10  }

11  play(){
12      form.hide();
13      textSize(30);
14      text("Game Start", 120, 100)
15      Player.getPlayerInfo();
16
17      if(allPlayers !== undefined){
18          var display_position = 130;
19          display_position+=20;
20          textSize(15);
21          text(allPlayers[plr].name + ": " + allPlayers[plr].distance, 120,display_position)
22      }
23  }
24
25  if(keyIsDown(UP_ARROW) && player.index !== null){
26      player.distance +=50
27      player.update();
28  }
29  }
30  }

```

¿Puedes escribir el código para hacer que el jugador actual aquí sea "rojo"?

El alumno agrega el código para cambiar el color de relleno, cuando el índice del jugador coincide con el jugador en la información.

```

play(){
    form.hide();
    textSize(30);
    text("Game Start", 120, 100)
    Player.getPlayerInfo();

    if(allPlayers !== undefined){
        var display_position = 130;
        for(var plr in allPlayers){
            if (plr === "player" + player.index)
                fill("red")
            else
                fill("black");



            display_position+=20;
            textSize(15);
            text(allPlayers[plr].name + ": " + allPlayers[plr].distance, 120,display_position)
        }
    }
}

```

	<p>OPCIONAL: Abrir el formulario y presionar jugar rápidamente,, arrojará un error porque la consulta de la base de datos para obtener playerCount aún no se ha completado.</p> <p>Para evitar este error, podemos configurar un oyente asíncrono .once() que obtendrá los datos playerCount solo una vez, y luego ejecutará getCount() para configurar un oyente permanente.</p> <p>La función asíncronica esperará el valor playerCount, y solo entonces, se creará y mostrará el formulario.</p> <p><i>Guíe al alumno para que agregue esto.</i></p>	<p>OPCIONAL: <i>El alumno agrega el código para esperar de forma asíncronica el valor playerCount.</i></p>
--	--	---

<pre> 5 var gameStateRef = database.ref('gameState'); 6 gameStateRef.on("value",function(data){ 7 gameState = data.val(); 8 }) 9 10 } 11 12 update(state){ 13 database.ref('/').update({ 14 gameState: state 15 }); 16 } 17 18 async start(){ 19 if(gameState === 0){ 20 player = new Player(); 21 var playerCountRef = await database.ref('playerCount').once("value"); 22 if(playerCountRef.exists()){ 23 playerCount = playerCountRef.val(); 24 player.getCount(); 25 } 26 form = new Form() 27 form.display(); 28 } 29 } 30 31 play(){ 32 form.hide(); 33 textSize(30); 34 text("Game Start", 120, 100) 35 Player.getPlayerInfo(); 36 37 if(allPlayers !== undefined){ 38 var display_position = 130; </pre>		
	<p>Finalmente, modifiquemos el archivo sketch para escribir condiciones para cambiar el estado del juego, y llamar a la función play().</p>	<p><i>El alumno escribe el código para actualizar el estado del juego, cuando el número de jugadores es 4, y llama a game.play() cuando el estado del juego es 1.</i></p>

<pre> 8 9 var form, player, game; 10 11 12 function setup(){ 13 canvas = createCanvas(400,400); 14 database = firebase.database(); 15 game = new Game(); 16 game.getState(); 17 game.start(); 18 } 19 20 21 function draw(){ 22 if(playerCount === 4){ 23 game.update(1); 24 } 25 if(gameState === 1){ 26 clear(); 27 game.play(); 28 } 29 } 30 </pre>	<p>Ejecutemos el código y veamos qué pasa.</p>	<p><i>El alumno ejecuta el código y verifica el resultado.</i></p>
<p>El Profesor Guía al Alumno para Dejar de Compartir Pantalla</p>		
<p><u>RETROALIMENTACIÓN</u></p> <ul style="list-style-type: none"> • Anime al alumno a tomar notas de reflexión en markdown. • Felicite al alumno por sus esfuerzos durante la clase. • Repasen el contenido aprendido. 		
<p>Paso 4: Conclusión (5 minutos)</p>	<p>¿Cómo te sentiste con la clase del día de hoy?</p>	<p>REA: variado</p>
	<p>Aunque parezca imposible de creer, ¡ya hemos hecho la mayor parte del trabajo duro en este juego!</p> <p>¡Todo lo que tenemos que hacer ahora, es dibujar el juego de autos usando sprites, y otras funciones de p5.play que ya conoces!</p>	<p>Asegúrese de haber regalado al menos 2 sombreros durante la clase por:</p> 

	<p>¡Felicidades por tu excelente trabajo!</p> <p>Haremos esto en la próxima clase.</p> <p>¡¡¡Excelente!!!</p> <p>Nos vemos pronto.</p>	 
<p><u>Nombre del proyecto: Juego MiCuestionario</u></p>	<p>Objetivo del proyecto:</p> <p>En la clase 37, has estructurado un juego de carreras de autos, has creado más propiedades y funciones en cada clase; has dado estados de juego y mostrado a todos los jugadores con sus puntuaciones de distancia recorrida.</p> <p>En este proyecto, aplicarás lo que has aprendido en la clase, para crear un juego de cuestionario para dos jugadores, y almacenar su respuesta a la pregunta del cuestionario en la base de datos.</p> <p>Historia:</p> <p>A Prakriti le encanta hacer cuestionarios. Y siempre trata de encontrar preguntas únicas y luego preguntarlas a diferentes personas. Pero ahora, está pensando en crear su propio juego de preguntas multijugador, en el que puede hacer una pregunta de tipo cuestionario a diferentes personas al mismo tiempo.</p> <p>¿Puedes ayudarla a crear el juego?</p>	<p><i>Los alumnos se involucran con el profesor en el proyecto.</i></p>

	<p>Estoy muy emocionado de ver la solución de tu proyecto y sé que lo harás realmente bien.</p> <p>¡Adiós!</p>	
<p style="text-align: center;">El profesor hace Clic en</p> <p style="text-align: right;">✕ Finalizar Clase</p>		
Actividades Adicionales	<p><i>Anime al alumno a escribir notas de reflexión en su diario de reflexión utilizando markdown.</i></p> <p>Úsalos como preguntas de orientación:</p> <ul style="list-style-type: none"> • ¿Qué pasó hoy? <ul style="list-style-type: none"> - Describe lo que pasó - Código que escribí • ¿Cómo me sentí después de la clase? • ¿Qué he aprendido sobre programación y desarrollo de juegos? • ¿Qué aspectos de la clase me ayudaron? ¿Qué encontré difícil? 	<p><i>El alumno usa el editor markdown para escribir su reflexión como un diario de reflexión.</i></p>

Actividad	Nombre de la Actividad	Enlaces
Actividad del Profesor 1	Código de Referencia	https://github.com/whitehatjr/CarRacingGame1.0 En Español: https://github.com/alejandraluna1/JuegoCarreras

		Autos1.0.git
Actividad del Alumno 1	Código de la clase anterior	https://github.com/whitehatjr/carRacingStage0.5 En Español: https://github.com/alejandraluna1/CarrerasDeAutosEtapa0.5.git
Solución del Proyecto	Juego MiCuestionario	https://priyapandey2020.github.io/MyQuiz-Game-Solution-37/ En Español: https://github.com/alejandraluna1/Solucion-C37-Juego-MiCuestionario.git

