

# **ANEXOS**

## **Anexos A**

### **Encuesta Realizada**

#### **Investigadores:**

Est. Narcisa de Jesús Araujo Pérez

Dra. Monica Rodas

Ing. Paola Ingavélez G.

Tiempo total estimado de todas las actividades:

#### **Nombre del participante:**

---

#### **Edad:**

---

#### **Posee Discapacidad: si/no, Especifique:**

---

#### **Profesión/Carrera:**

---

#### **Fecha:**

---

### **1.1 Formato de encuestas realizadas a participantes**

#### **Simulador Información Difícil**

Cuando termine, nuestro compañero entrevistador le preguntará y registrará los siguientes datos:

**1. El acceso al ejercitario le pareció:**

Muy fácil       Fácil       Regular       Difícil       Muy difícil

**2. Los íconos del ejercitario indican claramente la acción que realizan:**

Si       Parcialmente       No

**3. La información presentada sobre el ejercitario le pareció:**

Muy relevante       Algo relevante       Relevante       Poco relevante       Nada relevante

**4. El escenario de la simulación con respecto al ejercitario le pareció:**

Muy adecuado       Adecuado       Regular       O  
inadecuado       Muy inadecuado

**5. La interacción con la escena le pareció:**

Muy fácil       Fácil       Regular       Difícil       Muy difícil

**6. El acceso a las preferencias le pareció:**

Muy fácil       Fácil       Regular       Difícil       Muy difícil

**7. ¿Qué tan fácil fue cambiar el tipo de fuente?**

Muy fácil       Fácil       Regular       Difícil       Muy difícil

**8. ¿Qué tan fácil fue cambiar el fondo?**

Muy fácil       Fácil       Regular       Difícil       Muy difícil

**9. ¿Qué tan fácil fue cambiar el tamaño de letra?**

Muy fácil       Fácil       Regular       Difícil       Muy difícil

**10. ¿Qué tan fácil fue activar o desactivar el audio?**

O Muy fácil O Fácil O Regular O Difícil O Muy difícil

#### **11. Las opciones de preferencias le parecieron:**

O Muy completas      O Algo completas      O completas      O  
incompletas      O Muy incompletas

12. La navegación por la interfaz a través de ratón le pareció:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

13. La navegación por la interfaz a través de teclado le pareció:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

#### **14. Las actividades planteadas le parecieron:**

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

15. La retroalimentación proporcionada al terminar el ejercitario le pareció:

O Muy relevante O Algo relevante O Relevante O Poco relevante O Nada relevante

16. Con respecto al ejercitario que acaba de realizar piensa que:

O Si pudo realizarlo

O No pudo realizarlo

O No está seguro de haber finalizado

17. ¿En una escala del 0 al 10, cuál es su nivel de expectativa antes de ingresar al ejercitario?

0 0 0 0 0 0 0 0 0 0

**18. ¿En una escala del 0 al 10, cuál es su nivel de expectativa después de interactuar con el ejercitario?**

Muy bajo    1    2    3    4    5    6    7    8    9    10

Muy alto

**Argumenta las respuestas 17 y 18 sobre el simulador laboral:**

### **Simulador El Tiempo**

Cuando termine, nuestro compañero entrevistador le preguntará y registrará los siguientes datos:

**1. El acceso al ejercitario le pareció:**

   Muy fácil        Fácil        Regular        Difícil        Muy Difícil

**2. Los íconos del ejercitario indican claramente la acción que realizan:**

   Si        Parcialmente        No

**3. La información presentada sobre el ejercitario le pareció:**

   Muy relevante        Algo relevante        Relevante        Poco relevante        Nada relevante

**4. El escenario de la simulación con respecto al ejercitario le pareció:**

   Muy adecuado        Adecuado        Regular        O  
inadecuado        Muy inadecuado

**5. La interacción con la escena le pareció:**

   Muy fácil        Fácil        Regular        Difícil        Muy Difícil

**6. El acceso a las preferencias le pareció:**

   Muy fácil        Fácil        Regular        Difícil        Muy Difícil

**7. ¿Qué tan fácil fue cambiar el tipo de fuente?**

Muy fácil       Fácil       Regular       Difícil       Muy difícil

**8. ¿Qué tan fácil fue cambiar el fondo?**

Muy fácil       Fácil       Regular       Difícil       Muy difícil

**9. ¿Qué tan fácil fue cambiar el tamaño de letra?**

Muy fácil       Fácil       Regular       Difícil       Muy difícil

**10. ¿Qué tan fácil fue activar o desactivar el audio?**

Muy fácil       Fácil       Regular       Difícil       Muy difícil

**11. Las opciones de preferencias le parecieron:**

Muy completas       Algo completas       completas       incompletas       Muy incompletas

**12. La navegación por la interfaz a través de ratón le pareció:**

Muy fácil       Fácil       Regular       Difícil       Muy Difícil

**13. La navegación por la interfaz a través de teclado le pareció:**

Muy fácil       Fácil       Regular       Difícil       Muy Difícil

**14. Las actividades planteadas le parecieron:**

Muy fácil       Fácil       Regular       Difícil       Muy Difícil

**15. La asignación de valores correspondiente a jerarquizar cada actividad, le pareció:**

Muy fácil       Fácil       Regular       Difícil       Muy Difícil

**16. La asignación de valores correspondientes a delimitar tiempos de cada actividad, le pareció:**

Muy fácil       Fácil       Regular       Difícil       Muy Difícil

**17. La retroalimentación proporcionada al terminar el ejercitario le pareció:**

Muy relevante       Algo relevante       Relevante       Poco relevante       Nada relevante

**18. Con respecto al ejercitario que acaba de realizar piensa que:**

Si pudo realizarlo

No pudo realizarlo

No está seguro de haber finalizado

**19. ¿En una escala del 0 al 10, ¿cuál fue su nivel de expectativa antes de ingresar al ejercitario?**

Muy bajo      12      3      4      5      6      7      8      9      10      Muy alto

**20. ¿En una escala del 0 al 10, ¿cuál es su nivel de expectativa después de interactuar con el ejercitario?**

Muy bajo      12      3      4      5      6      7      8      9      10      Muy alto

**Argumenta las respuestas 19 y 20 sobre el simulador laboral:**

**Preguntas de Cierre (10 min)**

**1. ¿Considera que Los retos planteados fueron factibles de realizar?**

- 
- 
- 2. ¿Considera que el tiempo empleado para interactuar con los ejercitarios fue el adecuado?**
- 
- 

- 3. ¿Cuál fue la parte que le gustó de cada simulador laboral?**
- 
- 

- 4. ¿Cuál fue la parte que le gusto menos de cada simulador laboral?**
- 
- 

#### **Tareas de las personas incluidas en el estudio**

**Est. Narcisa de Jesús Araujo Pérez:** Explicación del proyecto Edutech en la Universidad y sus objetivos, presentación del proyecto desarrollado.

**Ing. Paola Ingavélez G:** Planificación y gestión de espacios y usuarios entrevistados.

**Dra. Mónica Rodas:** Validación y aprobación del plan de pruebas.

#### **Reporte de Evaluación**

Con los datos obtenidos se procederá a realizar las tabulaciones en base a las preguntas planteadas y su grado de cumplimiento. Adicionalmente se considera.

- Bitácoras de tiempos empleados por parte de los entrevistados.
- Errores detectados con descripción de cada uso.

- Comentarios identificados de nuestros usuarios.
- Fotografías

## Anexos B

### Protocolo de Evaluación



Fig 1 "Evaluación de Simuladores Laborales con Roberto Pacho"



Fig 2"Evaluación de Simuladores Laborales con Noe Ayavaca"



Fig 3"Evaluación de Simuladores Laborales con Michelle Peñalosa"



Fig 4"Evaluación de Simuladores Laborales con Fernando Deleg"



Fig 5"Evaluación de Simuladores Laborales con Fabian Armijos"



Fig 6 "Evaluación de Simuladores Laborales con Gabriel Chuchuca"

## Anexos C

### Modelado de Escenario, Personaje e

### Interfaz de Usuario Accesible

**Escenario Información Difícil**



Fig 7 "Vista entrada oficina rectorado"



Fig 8 "Vista parte interna de oficina"

### **Escenario El Tiempo**



Fig 9 "Vista escenario de Talento Humano"



Fig 10 "Vista escenario de Talento Humano"

### Personaje Simuladores Laborales

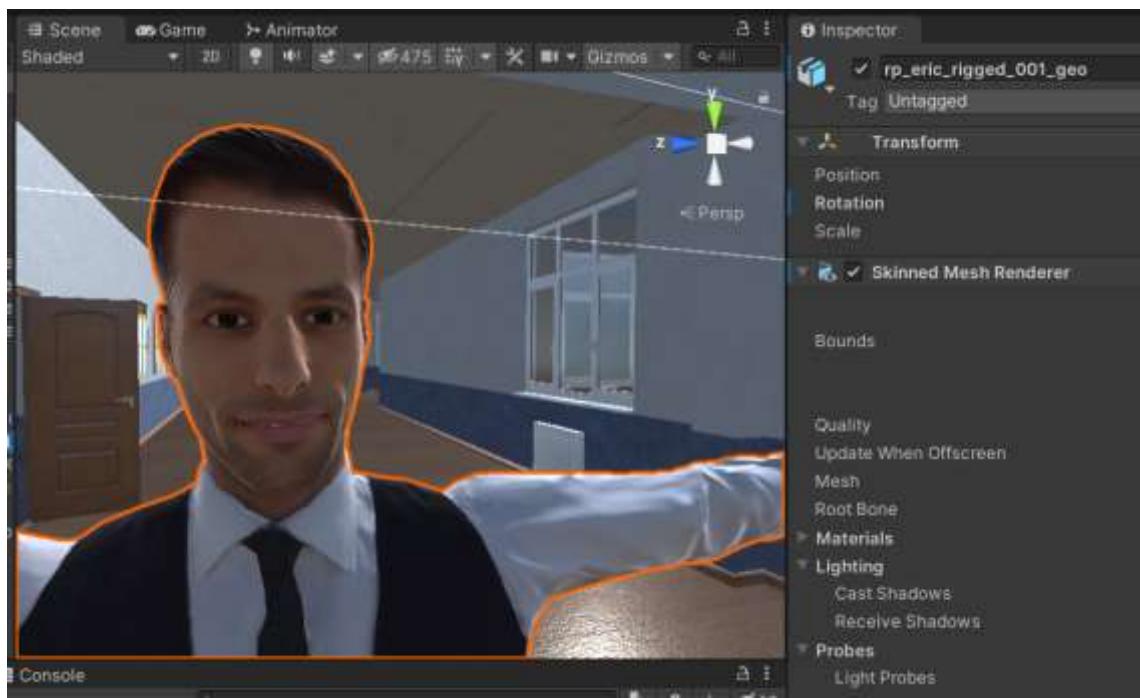


Fig 11 "Personaje utilizado para simuladores laborales"



Fig 12 "Diseño de interfaz de usuario de simuladores laborales"



Fig 13 "Diseño de barra de instrucciones de simuladores laborales"

## Anexos D

### Animación de Personaje e interfaz de

### usuario

**Animación Personaje Simuladores Laborales**  
**Animación Personaje Simulador Laboral Información Difícil**



Fig 14 "Animación del personaje en el escenario del Simulador Laboral Información Difícil"



Fig 15 "Animación del rostro del personaje del Simulador Laboral Información Difícil"

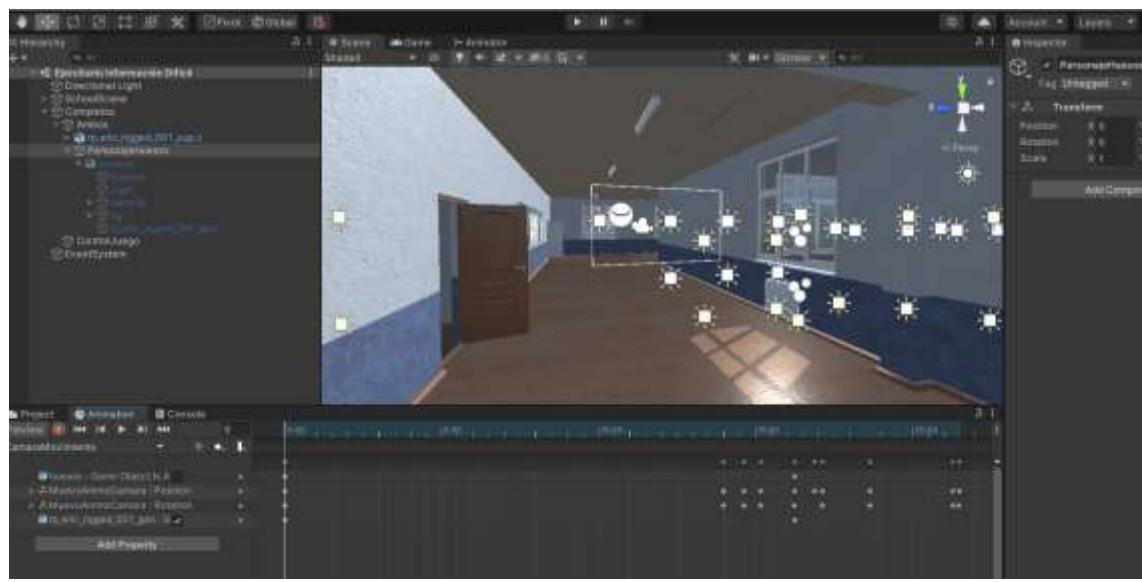


Fig 16"Animación vista de cámara en primera persona correspondiente al Simulador Laboral Información Difícil"

### Animación personaje Simulador Laboral El Tiempo



Fig 17"Animación de personaje en el escenario del Simulador Laboral El Tiempo"



Fig 18"Animación de vista de cámara en primera persona correspondiente al personaje del Simulador Laboral El Tiempo"

### Animación de Interfaz de usuario



Fig 19"Animación creada dentro del Animator Controller Canvas correspondiente a la interfaz de usuario del Simulador Laboral Información Difícil "

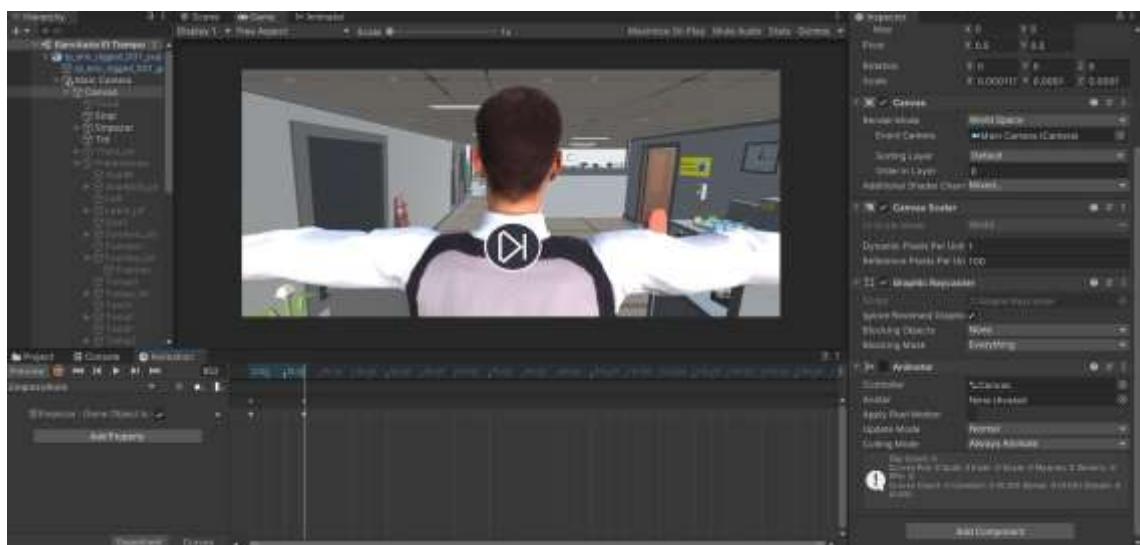


Fig 20 "Animación creada dentro del Animator Controller Canvas correspondiente a la interfaz de usuario del Simulador Laboral el Tiempo"

## Anexos E

### Simuladores Laborales 3D en ejecución

#### Simulador Información Difícil en Ejecución



Fig 21 "Vista inicial de ejecución de simulador laboral Información Difícil"



Fig 22 "Vista ingreso de personaje a oficina"

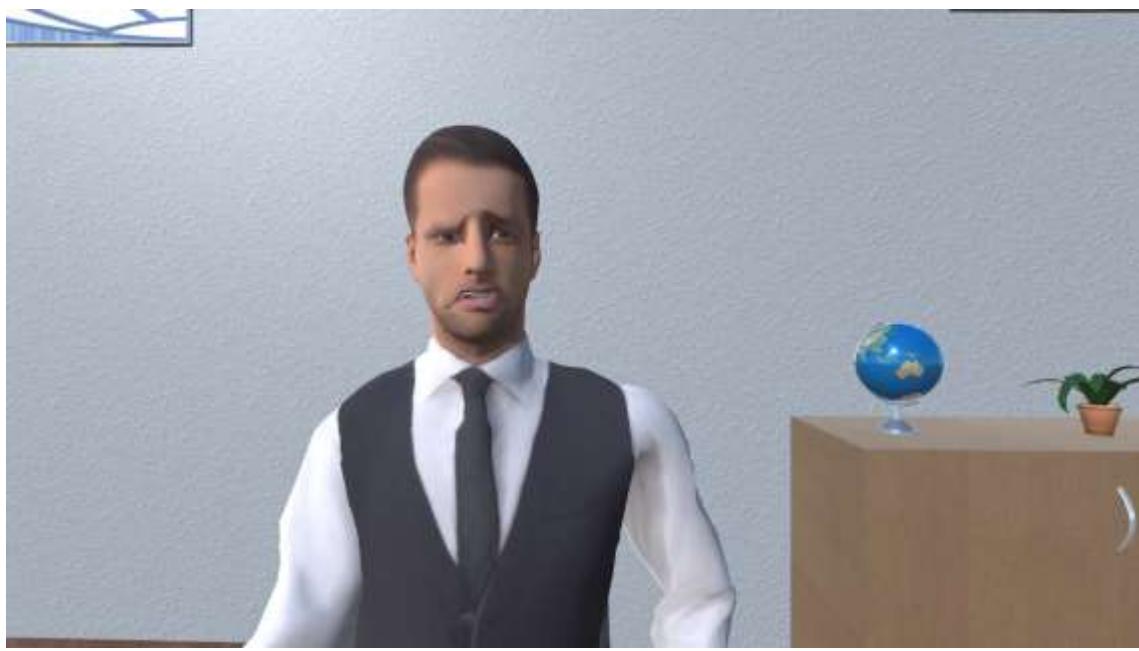


Fig 23 "Vista expresión de preocupación del personaje a causa de la difícil decisión que debe tomar "

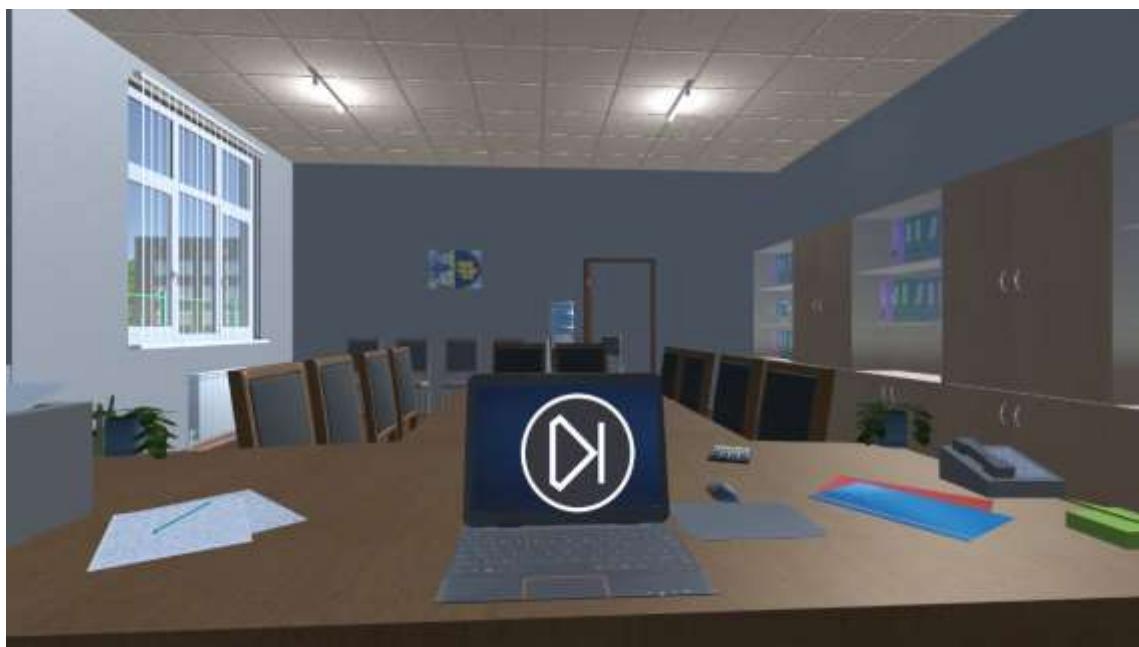


Fig 24 "Vista panel en donde se muestra el botón empezar del simulador laboral "



Fig 25 "Vista de interacción de personaje con el escenario, en el cual puede mover la vista de la cámara en primera persona e interactuar con objetos del mismo"



Fig 26 "Vista en donde el personaje interactúa con el botón empezar"



Fig 27"Vista del primer panel en donde se presentan instrucciones para el participante referente al ejercitario (ejercicio) planteado"



Fig 28"Vista interacción de participante con el botón siguiente mediante el teclado"

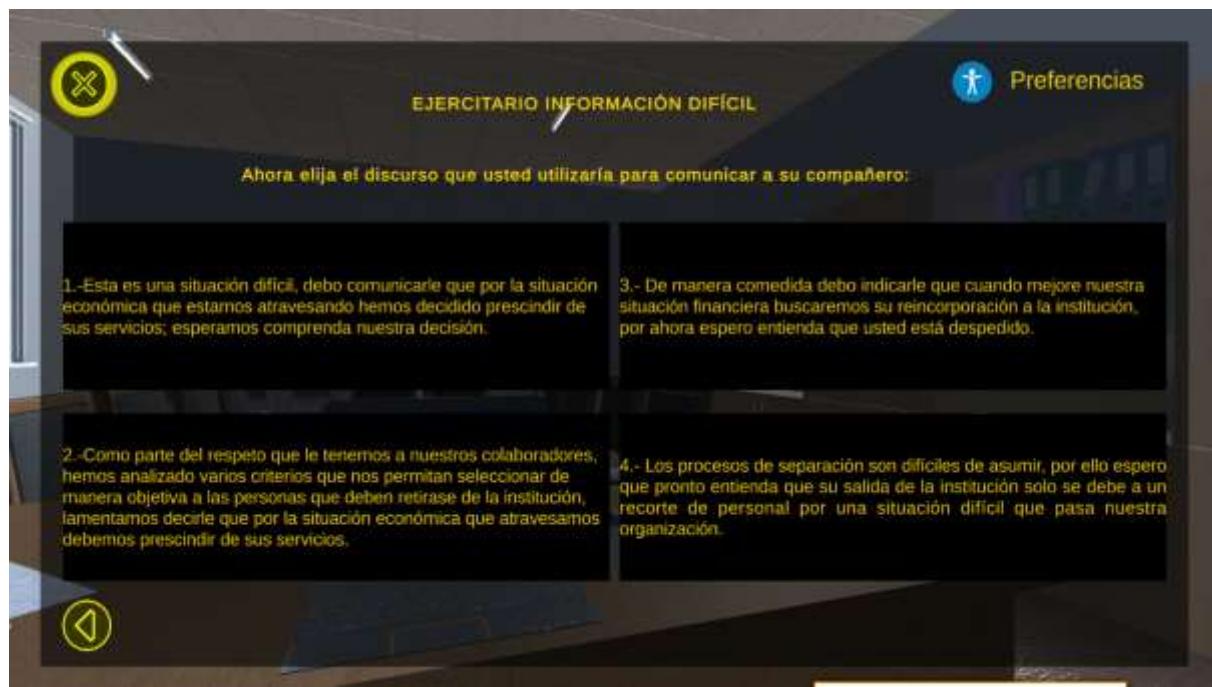


Fig 29 "Vista panel de actividades presentadas al participante"



Fig 30 "Vista de interacción del participante con las actividades planteadas mediante teclado"

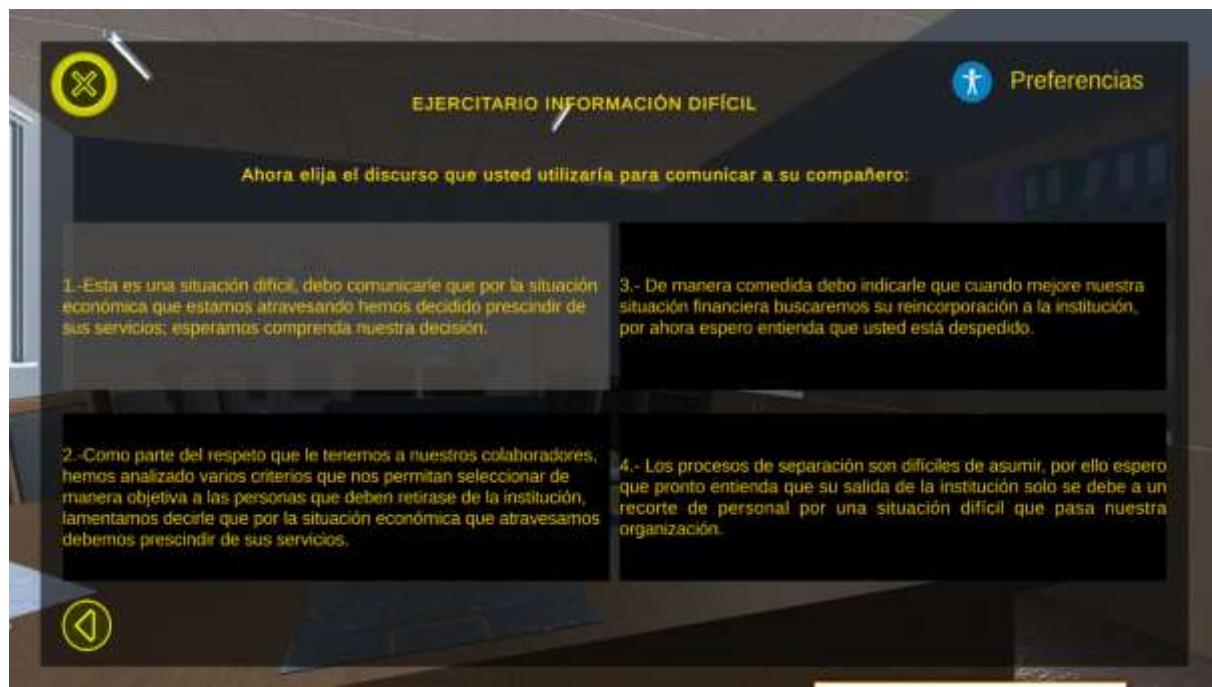


Fig 31 "Vista de interacción del participante con las actividades presentadas mediante el mouse"



Fig 32 "Vista en donde el participante revisa la calificación obtenida luego de haber finalizado la interacción con las actividades planteadas; cuando el participante llegue a este panel ya no podrá regresar"



Fig 33 "Vista final del simulador laboral en donde el participante revisa la retroalimentación referente al ejercitario (ejercicio) planteado"



Fig 34 "Vista de interacción de participante con el botón volver a menú principal"

- **Interacción del participante con las distintas opciones de accesibilidad del simulador laboral**



Fig 35 "Vista de participante interactuando con la barra de preferencias del simulador laboral mediante el teclado"



Fig 36 "Vista en donde el participante selecciona la fuente de texto Arial Bold"



Fig 37"Vista en donde el participante selecciona la fuente de texto Lato"



Fig 38"Vista en donde el participante selecciona la fuente Dyslexic"

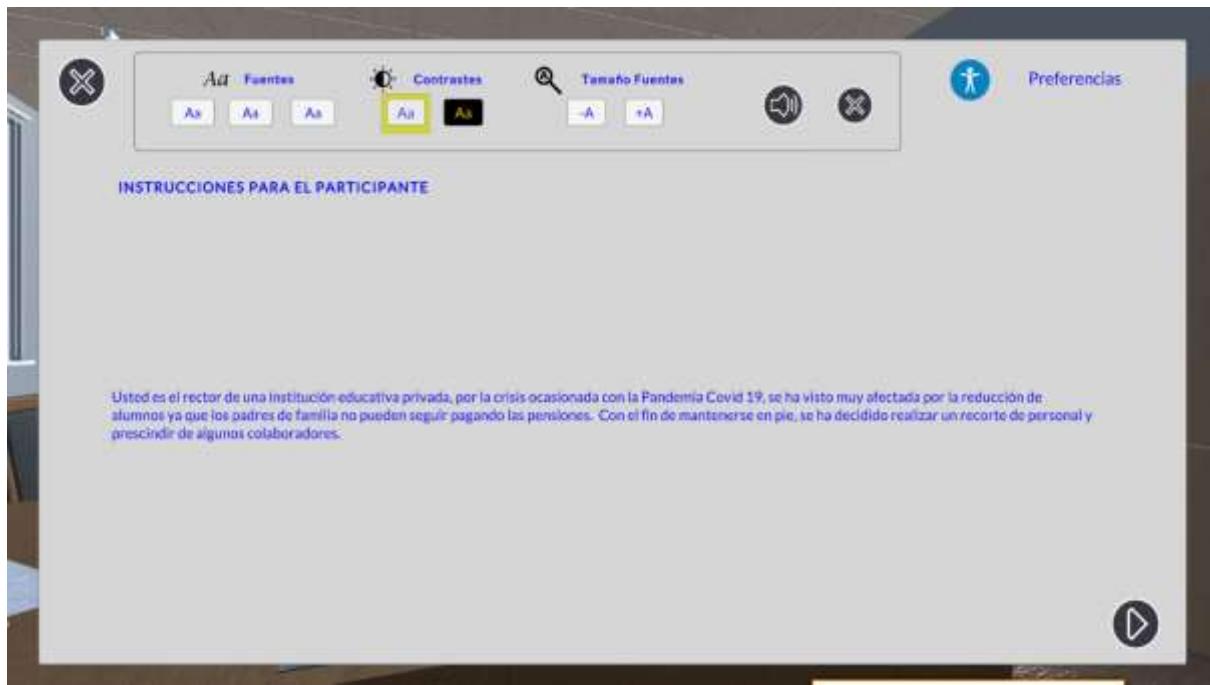


Fig 39 "Vista en donde el participante selecciona un contraste claro"

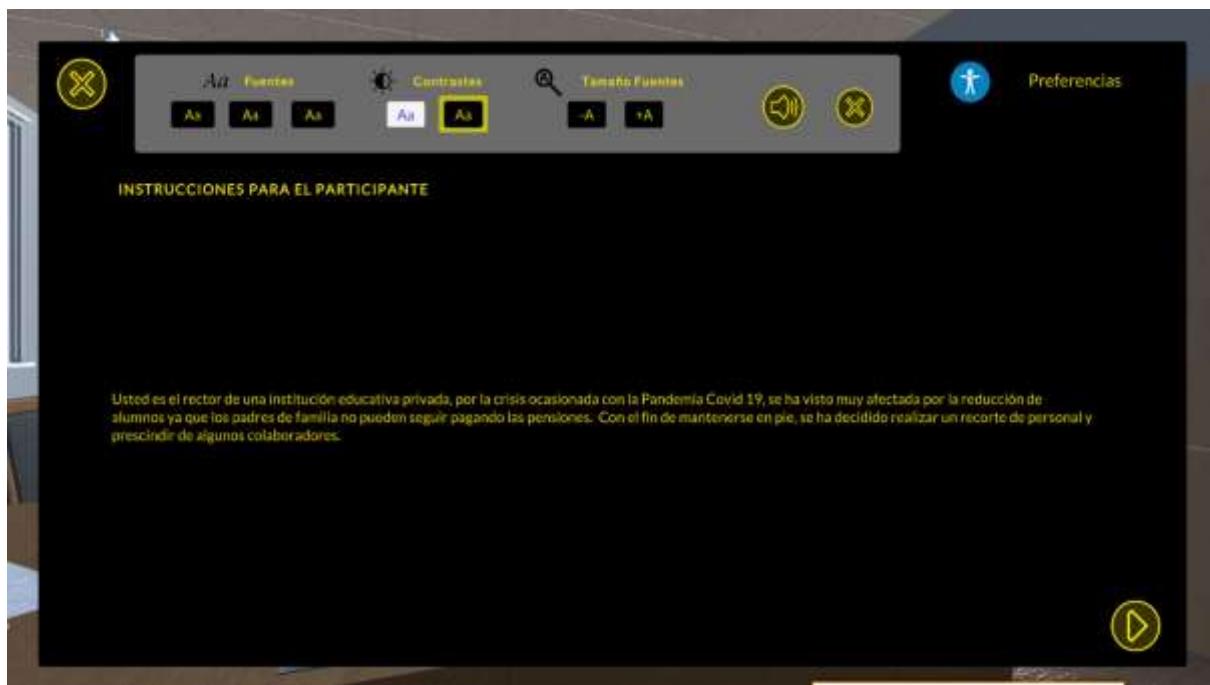


Fig 40 "Vista en donde el participante selecciona un contraste oscuro"



Fig 41 "Vista en donde participante presiona el botón de reducir el tamaño de texto"



Fig 42 "Vista en donde el participante presiona el botón para aumentar el tamaño de texto"



Fig 43 "Vista en donde el participante desactiva la descripción de audios"



Fig 44 "Vista en donde el participante cierra la barra de preferencias al presionar el botón salir de preferencias"

## Simulador Laboral El Tiempo en Ejecución

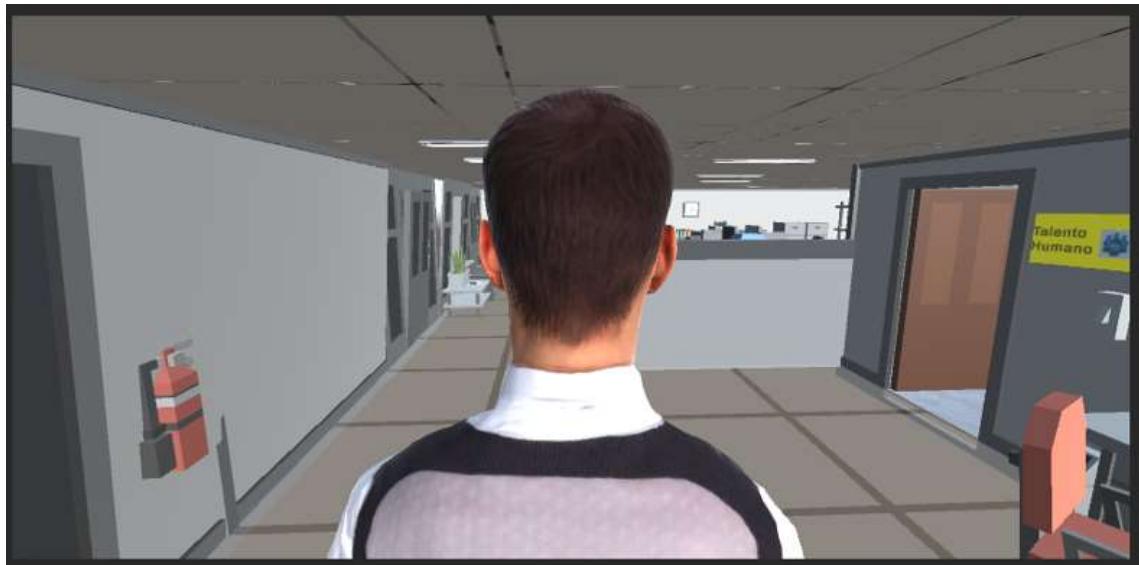


Fig 45"Vista inicial de ejecución de simulador laboral El Tiempo"



Fig 46"Vista de ingreso de personaje a oficina"



Fig 47"Vista personaje acercándose a su puesto de trabajo"



Fig 48"Vista panel en donde se muestra el botón empezar del simulador laboral "



Fig 49 "Vista de interacción de personaje con el escenario, en el cual puede mover la vista de la cámara en primera persona e interactuar con objetos del mismo"



Fig 50 "Vista interacción de personaje con botón empezar, en donde puede hacerlo mediante el teclado o el mouse"



Fig 51"Vista del primer panel en donde se presentan instrucciones para el participante referente al ejercitario (ejercicio) planteado"

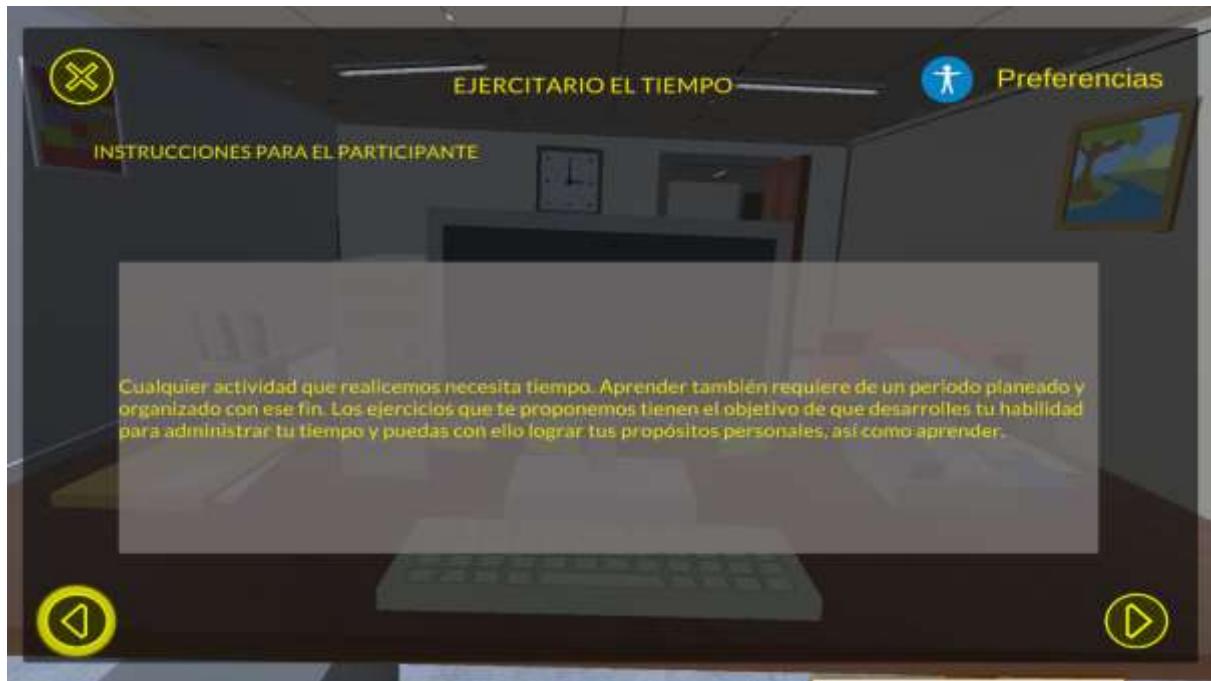


Fig 52"Vista en donde el participante interactúa con el panel de instrucciones e1 mediante el mouse"



Fig 53"Vista de segundo panel de instrucciones, en donde el participante interactúa con el botón siguiente mediante el mouse "

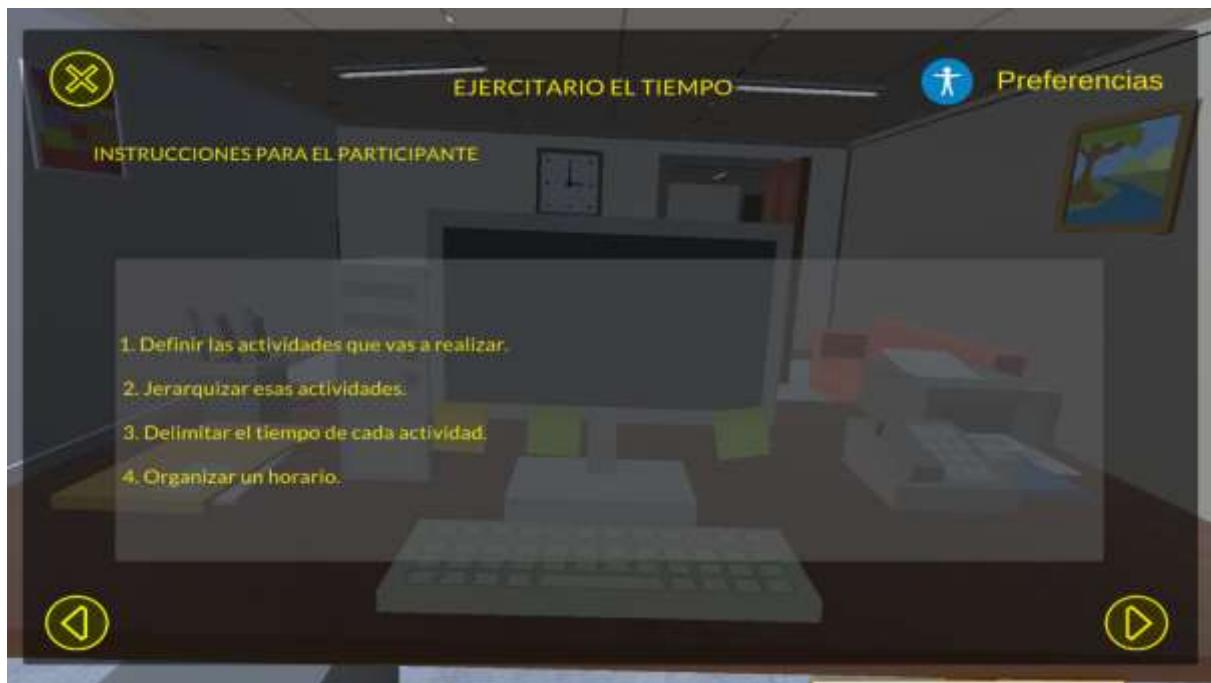


Fig 54"Vista de tercer panel de instrucciones en donde el participante interactúa con el teclado"

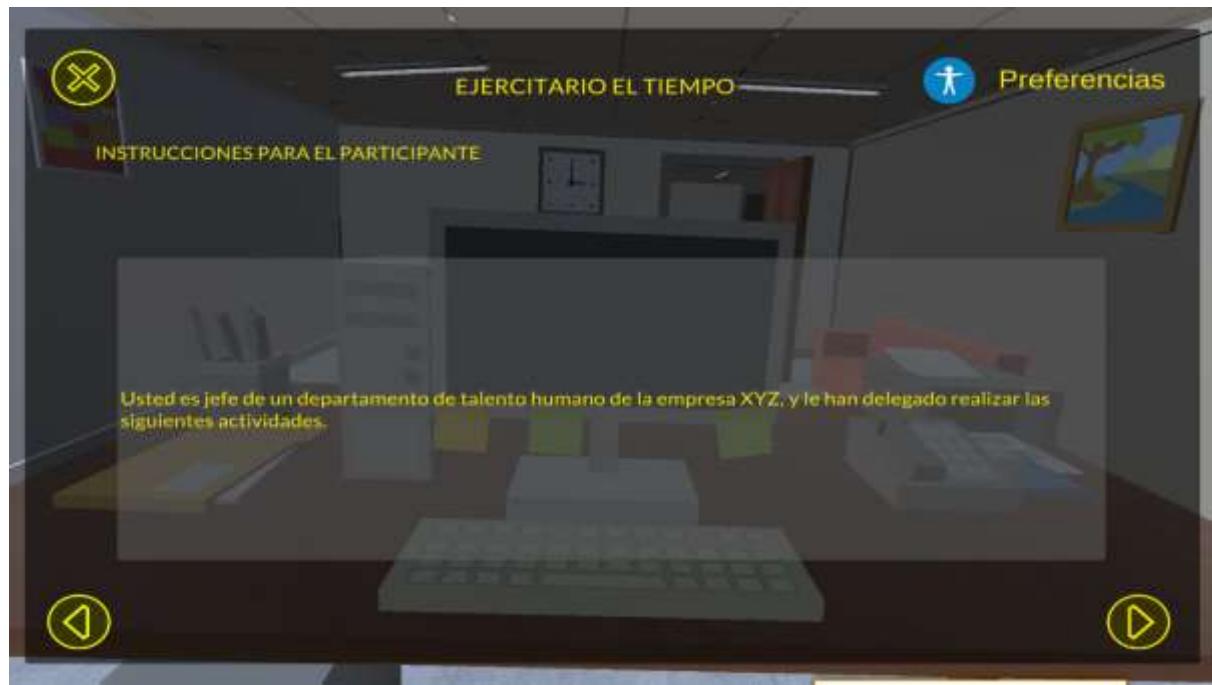


Fig 55"Vista de cuarto panel de instrucciones, en donde le participante interactúa con el teclado"

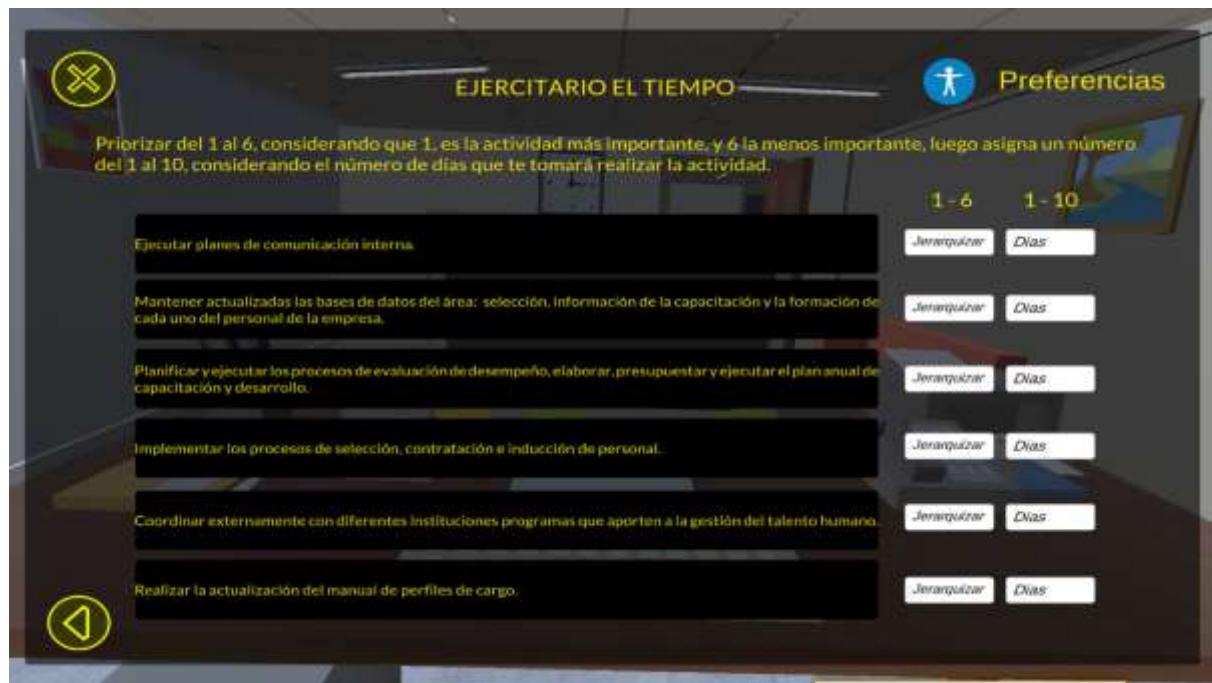


Fig 56"Vista panel en donde se indica las actividades planteadas con las que deberá interactuar el participante"

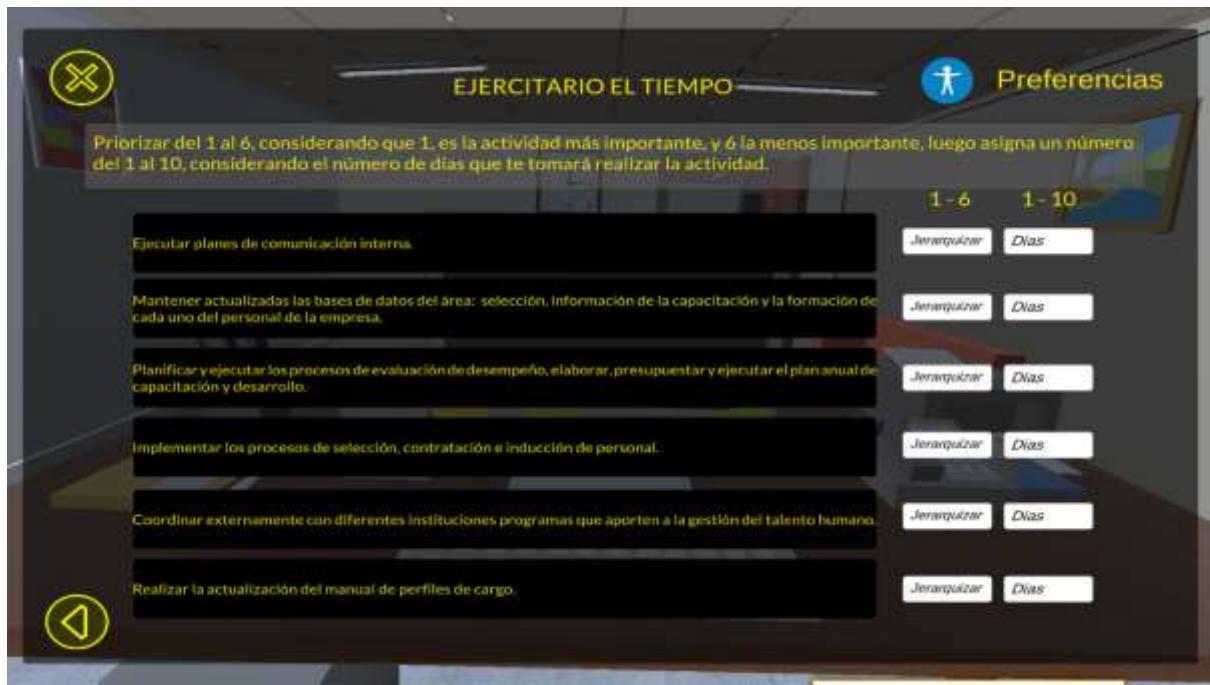


Fig 57"Vista de interacción del participante con panel de actividades mediante el teclado"

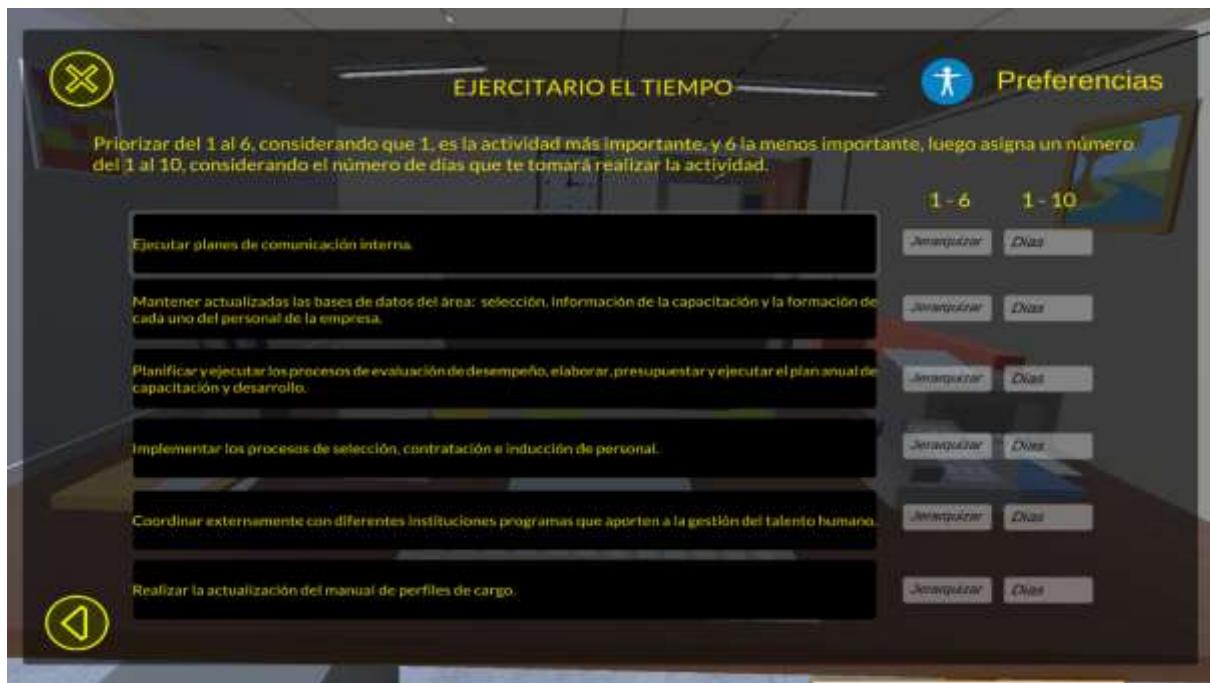


Fig 58"Vista en donde se inactivan los campos de texto de las respuestas al interactuar con el teclado"

**EJERCITARIO EL TIEMPO**

Priorizar del 1 al 6, considerando que 1 es la actividad más importante, y 6 la menos importante, luego asigna un número del 1 al 10, considerando el número de días que te tomará realizar la actividad.

Ejecutar planes de comunicación interna.	1 - 6	1 - 10
Mantener actualizados las bases de datos del área: selección, información de la capacitación y la formación de cada uno del personal de la empresa.	Jerarquizar	Días
Planificar y ejecutar los procesos de evaluación de desempeño, elaborar, presupuestar y ejecutar el plan anual de capacitación y desarrollo.	Jerarquizar	Días
Implementar los procesos de selección, contratación e inducción de personal.	Jerarquizar	Días
Coordinar externamente con diferentes instituciones programas que aporten a la gestión del talento humano.	Jerarquizar	Días
Realizar la actualización del manual de perfiles de cargo.	Jerarquizar	Días

**Preferencias**

Fig 59"Vista de ingreso de respuestas del participante"

**EJERCITARIO EL TIEMPO**

Priorizar del 1 al 6, considerando que 1 es la actividad más importante, y 6 la menos importante, luego asigna un número del 1 al 10, considerando el número de días que te tomará realizar la actividad.

Ejecutar planes de comunicación interna.	1 - 6	1 - 10
Mantener actualizados las bases de datos del área: selección, información de la capacitación y la formación de cada uno del personal de la empresa.	2	8
Planificar y ejecutar los procesos de evaluación de desempeño, elaborar, presupuestar y ejecutar el plan anual de capacitación y desarrollo.	3	6
Implementar los procesos de selección, contratación e inducción de personal.	6	6
Coordinar externamente con diferentes instituciones programas que aporten a la gestión del talento humano.	5	6
Realizar la actualización del manual de perfiles de cargo.	4	10

**Preferencias**

Fig 60"Vista en donde en participante ha completado todas las respuestas y por ello se activa el botón siguiente"

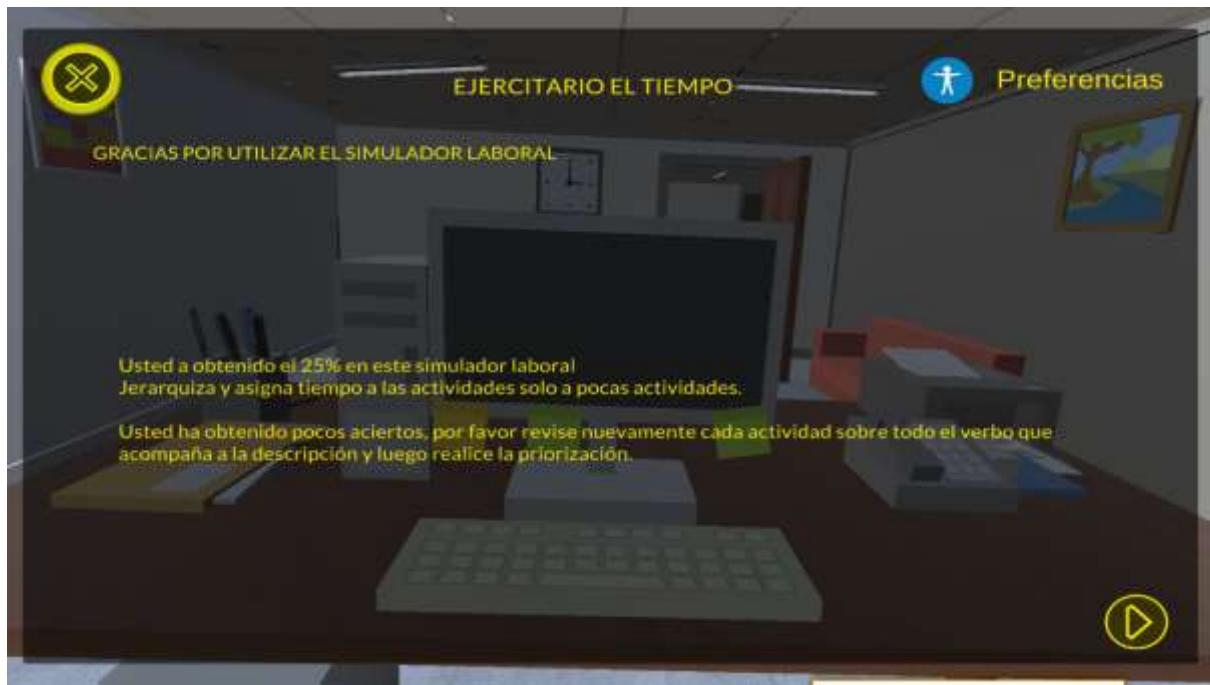


Fig 61 "Vista panel de calificación luego de que el participante haya finalizado las actividades planteadas"



Fig 62 "Vista panel de retroalimentación"



Fig 63"Vista en donde el participante interactúa con el botón para volver al menú principal"

#### Interacción del participante con las distintas opciones de accesibilidad del simulador laboral

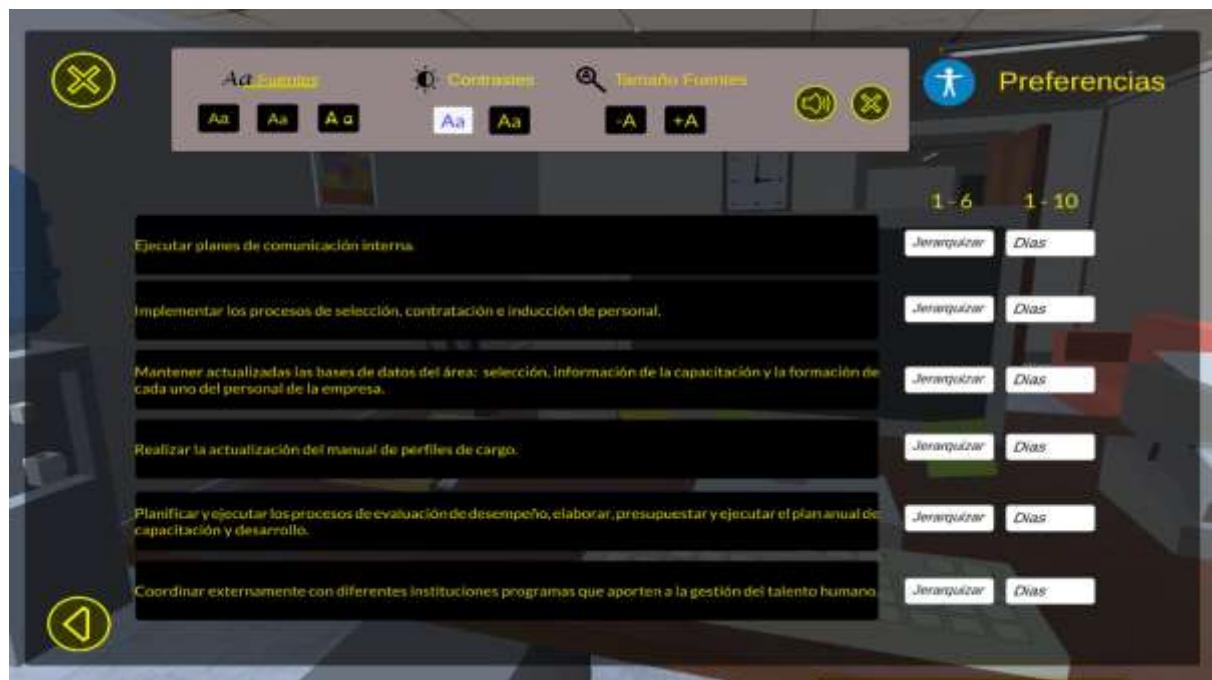


Fig 64" Vista de participante en la barra de preferencias del simulador laboral mediante el mouse"



Fig 65"Vista en donde el participante interactúa con el teclado y selecciona la fuente de texto Arial Bold"



Fig 66"Vista en donde el participante interactúa con el teclado y selecciona la fuente de texto Lato"

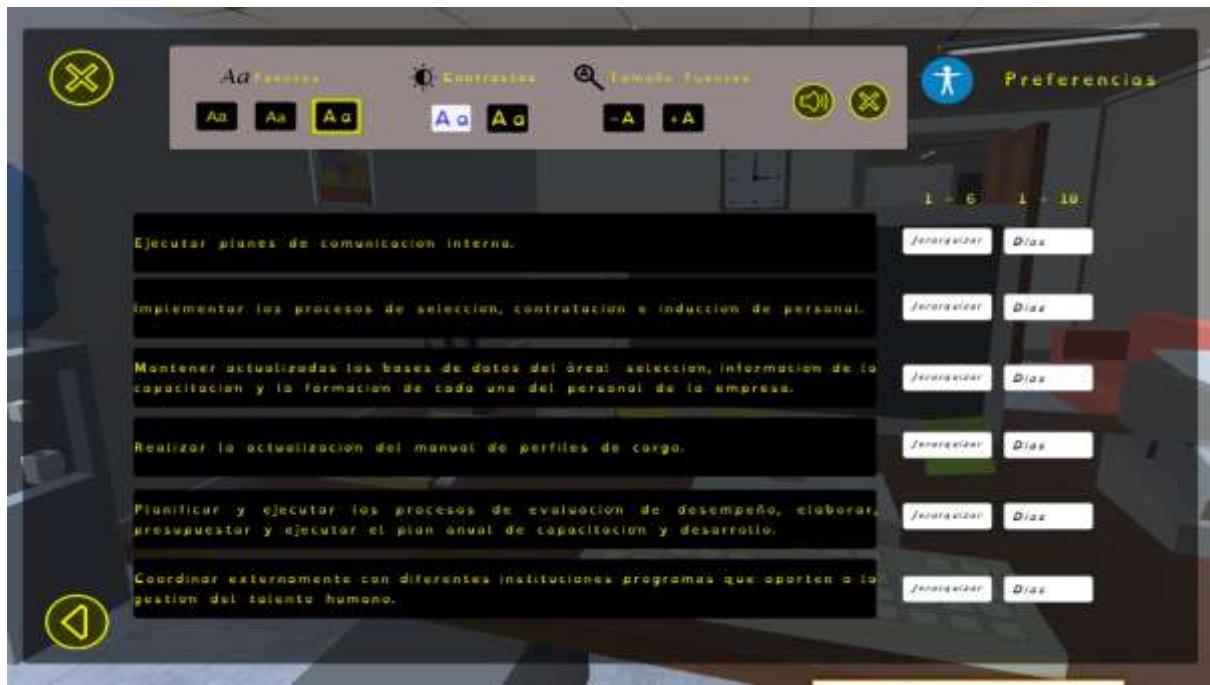


Fig 67"Vista en donde el participante interactúa con el teclado y selecciona la fuente de texto Dyslexic"

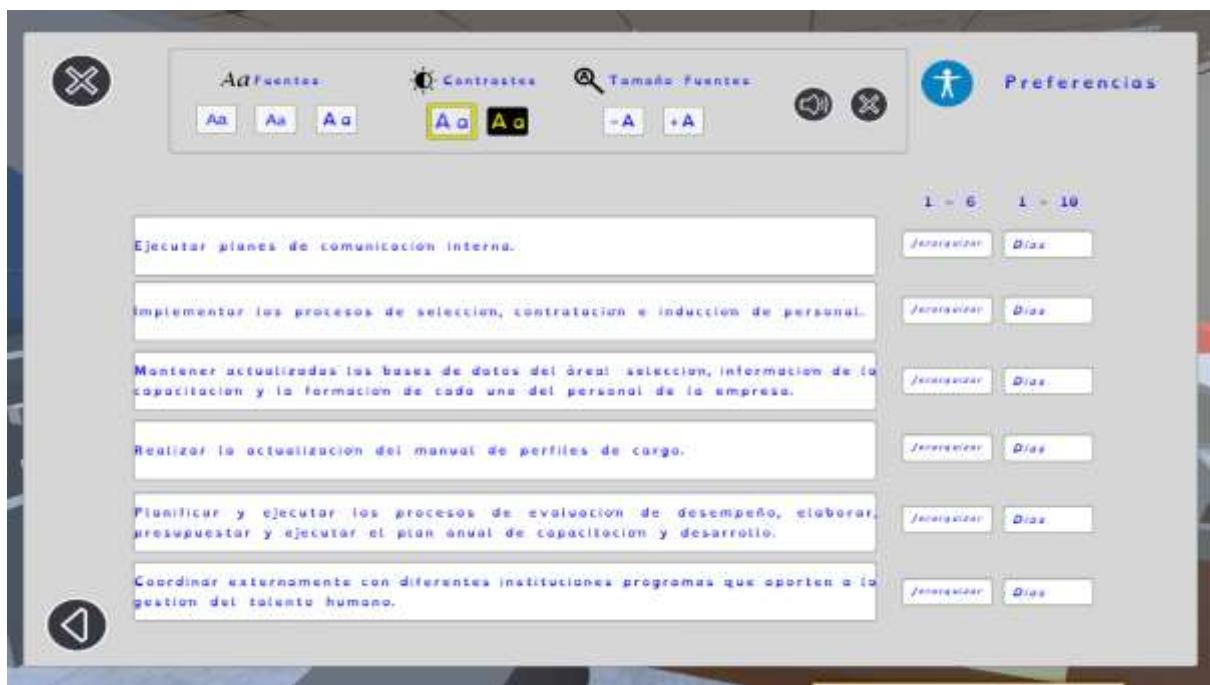


Fig 68"Vista en donde el participante interactúa con el teclado y selecciona un contraste claro"



Fig 69 "Vista en donde el participante interactúa con el teclado y selecciona un contraste oscuro"

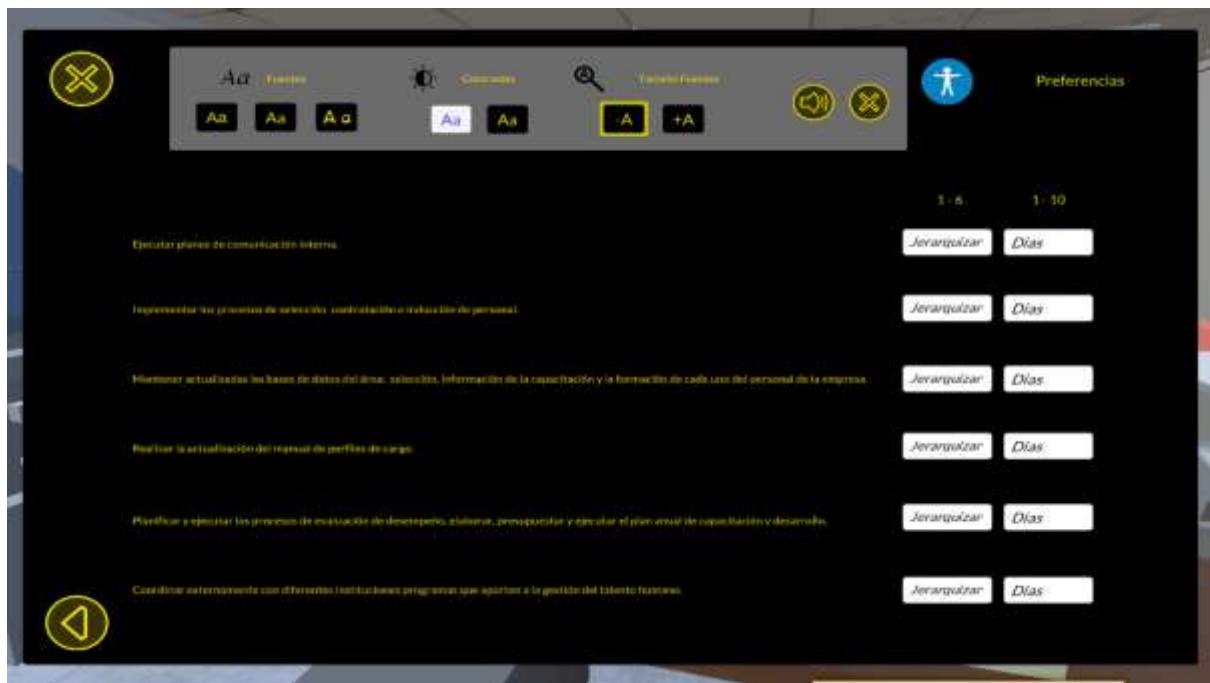


Fig 70 "Vista en donde el participante interactúa con el teclado y selecciona el botón de reducir el tamaño de texto"

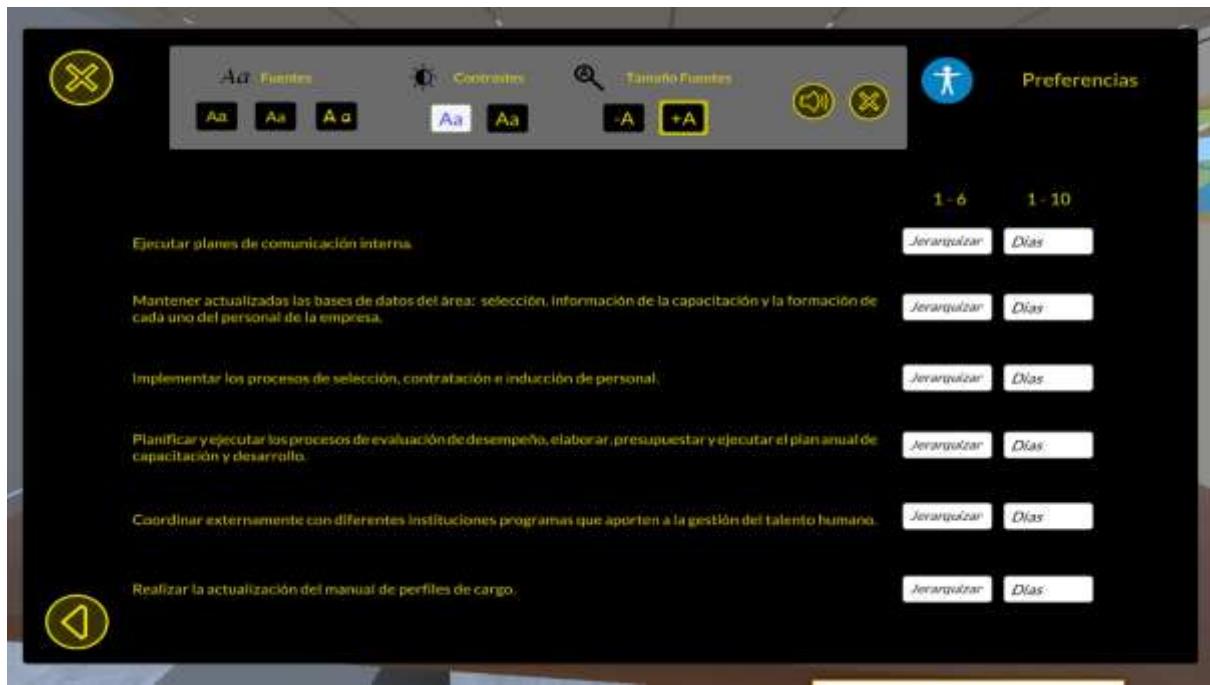


Fig 71 "Vista en donde el participante interactúa con el teclado y selecciona el botón de aumentar el tamaño de texto"

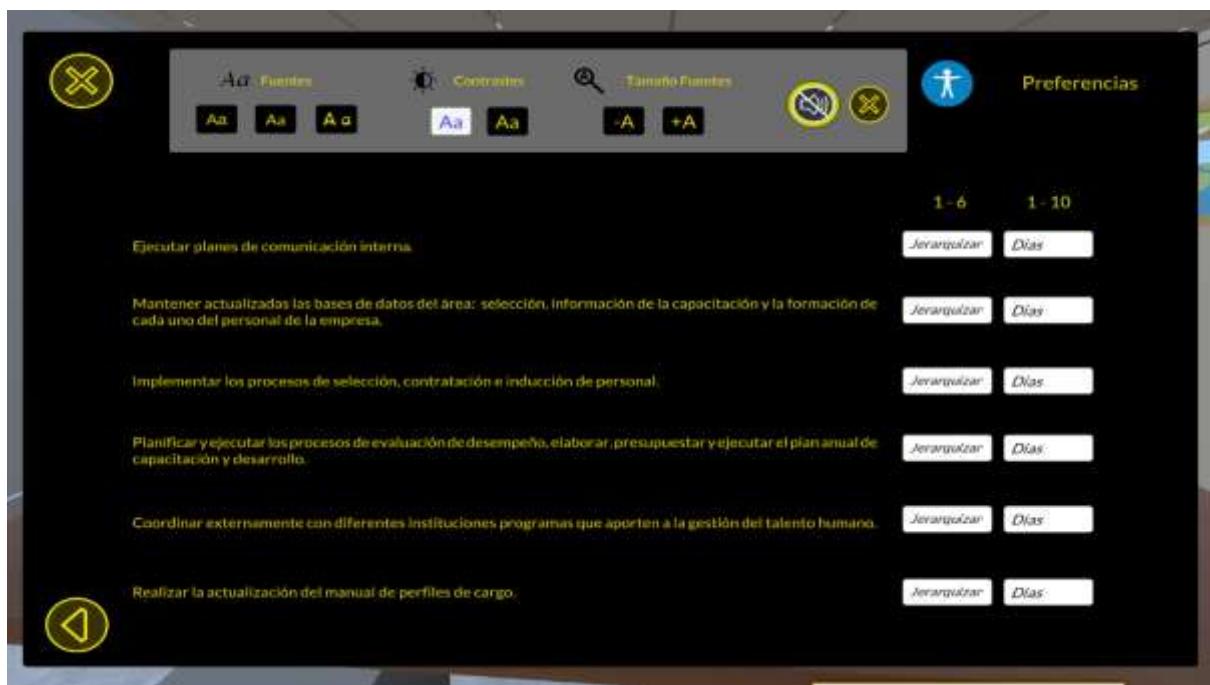


Fig 72 "Vista en donde el participante interactúa con el teclado y selecciona el botón desactivar el audio"

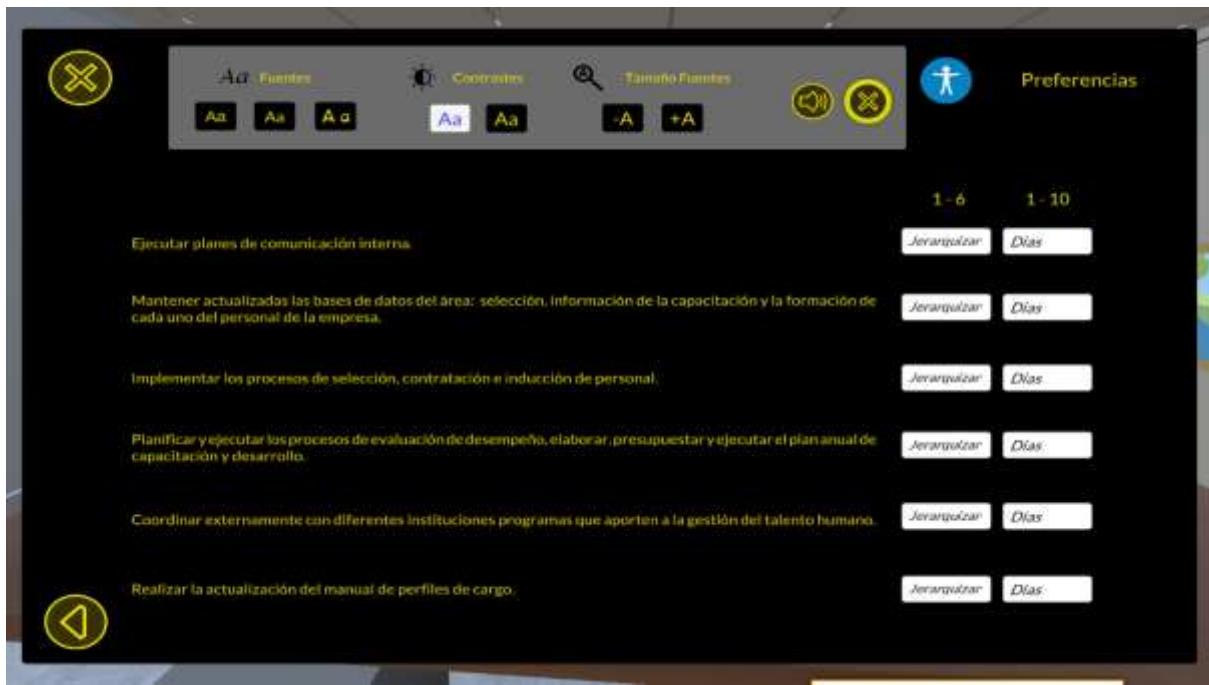


Fig 73 "Vista en donde el participante interactúa con el mouse y selecciona el botón salir de la barra de preferencias"

## Anexos F

# Manual Técnico

## 1. Funcionamiento de simuladores laborales

### 1.1 Animación de Personaje

Luego de agregar el personaje a la escena del proyecto procedemos a crear la respectiva animación, desde nuestro personaje agregamos un componente animator controller desde la ventana inspector, luego desde la ventana Project nos dirigimos a la carpeta en donde vamos a crear nuestras animaciones, en este caso nos dirigimos a Assets/Animations una vez dentro damos clic derecho create/animator controller le asignamos un nombre y procedemos a crear nuestra animación.

Para las animaciones del personaje se hace uso de varias animaciones descargadas de la página mixamo, dichas animaciones las agregamos a nuestro animator que creamos *CaminandoInicial*.

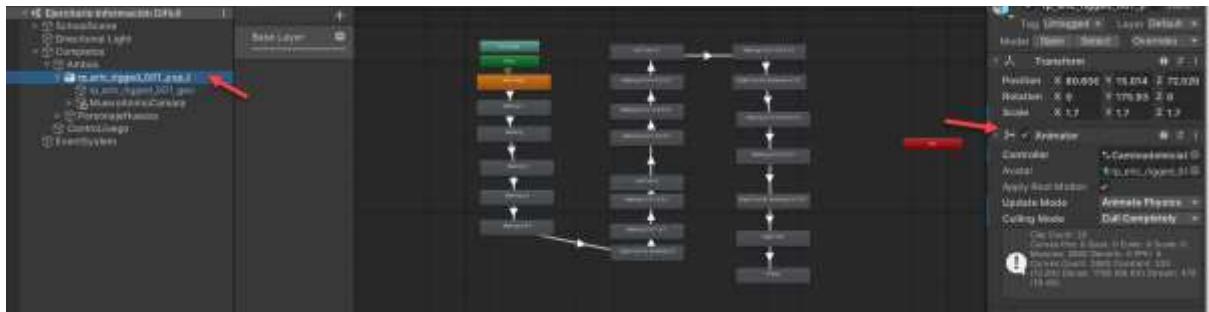


Fig 74 "Animación inicial de personaje en animator controller de Unity"

Se crea otra animación en donde el personaje cambia de expresión facial al llegar a su lugar de trabajo. Desde el gameobject *Ambos* se crea un *animator controller*, creamos un clip de animación modificamos las propiedades *is Active* del prefab de los huesos del personaje; la propiedad *transform* del gameobject *MueveAnimoCamara* en donde se modifica la *posición* y *rotación* de la cámara y la propiedad *IsActive* del personaje. Dichas propiedades las modificamos en la línea de tiempo según a las necesidades del proyecto.

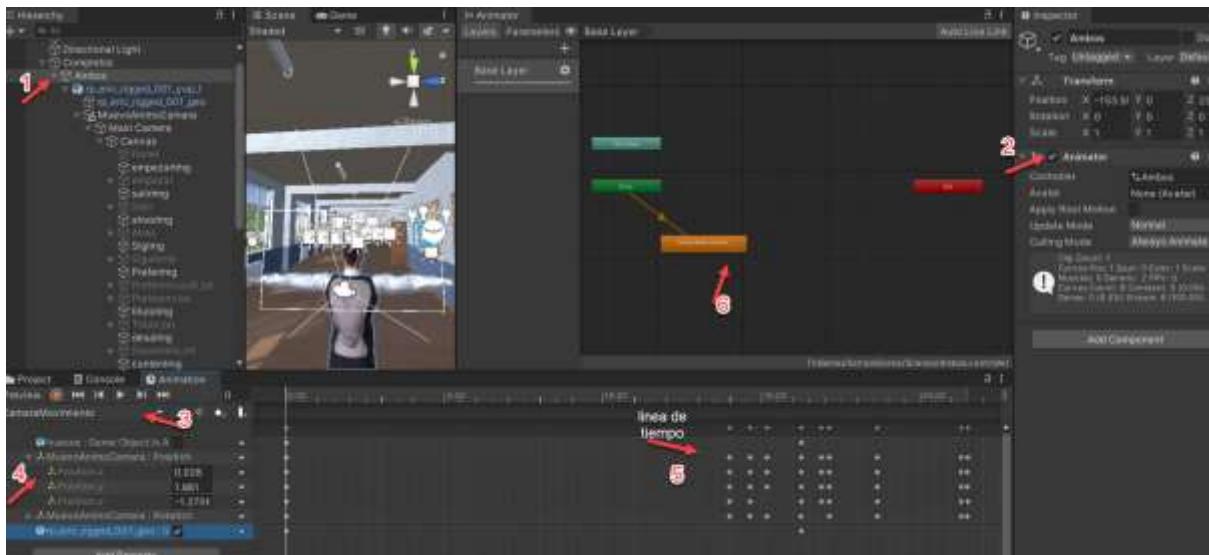


Fig 75 "Animación de personaje en la oficina"

## 1.2 Animación Canvas de Interfaz de usuario

Luego del diseño de la interfaz de usuario se procedió a darle sentido al mismo, se creó una animación para que luego de que termine la animación del personaje aparezca en pantalla el canvas es decir la interfaz de usuario creada.

A continuación desde nuestro Canvas agregamos un componente animator controller llamado *Canvas*, desde la ventana *Animation* creamos una nueva animación damos clic en *create new clip* y se asignamos un nombre a la animación en este caso se llama *Empezar*, luego damos clic en *Add Property* nos aparece un menú del cual seleccionamos el botón a animar y finalmente seleccionamos la propiedad *is Active*. La animación que se hace es modificar la linea de tiempo de la propiedad *Is Active*, que al ejecutar el proyecto pase 21 segundos(tiempo en que termina la animación del personaje) para que el botón active.

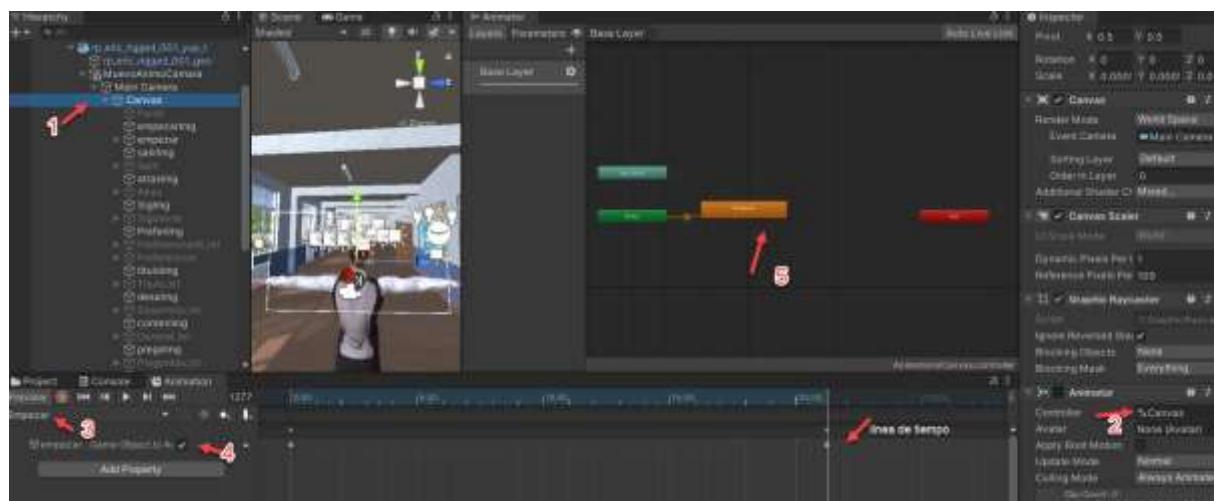
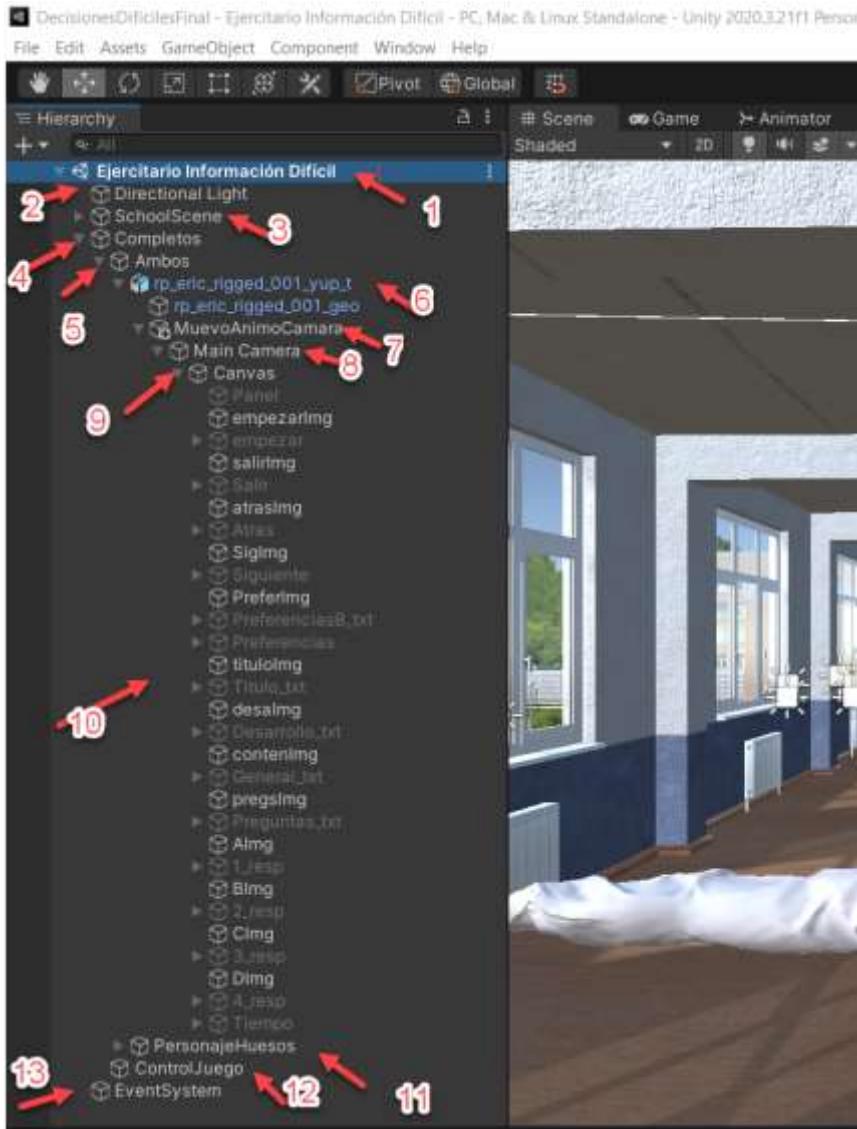


Fig 76 "Animación de canvas usando un animator controller"

### 1.3 Vista de Jerarquía del Proyecto

En la ventana *Inspector* asociamos los componentes creados con las variables declaradas, para ello arrastramos cada componente hacia la variable con la que se desea asociar.



2. Fig 77"Vista jerarquía del proyecto la cual contiene todos los objetos utilizados"

1. Escena del proyecto.
2. Iluminación de escena.
3. Escenario de simulador laboral.
4. Se crea el gameobject *Completos* y dentro de este se crea dos gameobjects.

5. El gameobject *Ambos* contiene el personaje de la escena, dentro del personaje arrastramos la cámara principal para que esta se mueva con el personaje cuando inicie la animación.
6. Personaje de la escena.
7. Gameobject que contiene la cámara principal.
8. Cámara principal de la escena.
9. Se crea un Canvas el cual contiene todos los componentes de la interfaz de usuario.
10. Corresponde a todos los componentes de la interfaz de usuario a simular.
11. Personaje que se muestra después que termina la animación del personaje inicial.
12. Gameobject que contiene el script principal *GeneralEmpezar.cs* y el script *WebData.cs* que consume los servicios web.
13. Es la manera en la que se envían los eventos a objetos basados en input (mouse, teclado).

## 1.4. Programación Simulador Información Difícil.

### 1.4.1 Script GeneralEmpezar.cs

```
using System;
using System.Collections;
using System.Collections.Generic; //Librería para manipular con listas
using UnityEngine;
using System.Linq; //Librería para realizar consultas
using System.Text; //Librería para manipular texto
using UnityEngine.UI; //Librería para acceder a las propiedades de los elementos de la interfaz de usuario.
using System.IO; //Librería para escribir y leer datos.
using System.Threading.Tasks;
using UnityEngine.Events;
using TMPro; //Importamos la librería TextMeshPro.
using Random = UnityEngine.Random;
using UnityEngine.Networking; //Importamos la librería para la comunicación con el servidor web.
```

Fig 78 "Librerías utilizadas"

#### 1.2.1.1 Declaración de variables y relación con objetos de interfaz de Unity

Desde la ventana de jerarquía en el gameobject ControlJuego una vez posicionados aquí nos dirigimos a la ventana inspector y procedemos a relacionar los objetos con las variables declaradas para ello arrastramos cada objeto y lo colocamos en la variable correspondiente.

```
// Se declaran las variables públicas a las cuales las asociamos a los gameobjects correspondientes.
public GameObject juego, botSalir, botPref, botSig, botAtras, tit, detail, conten, pregu, botEmpe, personaje, personajeHuesos;
//Para mostrar texto declararemos variables públicas de tipo TextMeshProUGUI y asociaremos dichas variables
//con los gameobjects creados en el paso anterior las cuales contienen los componentes TextMeshProUGUI
public TextMeshProUGUI textoEmpleo, textoTit, textPreg, textodetail, textoconten, tiempoText, textoSig;
//Se declara las variables públicas de tipo Image las cuales las asociamos al botón empezar y al botón siguiente.
public Image empleo, nextB;
```

Fig 79 "Fragmento de código de declaración de variables"

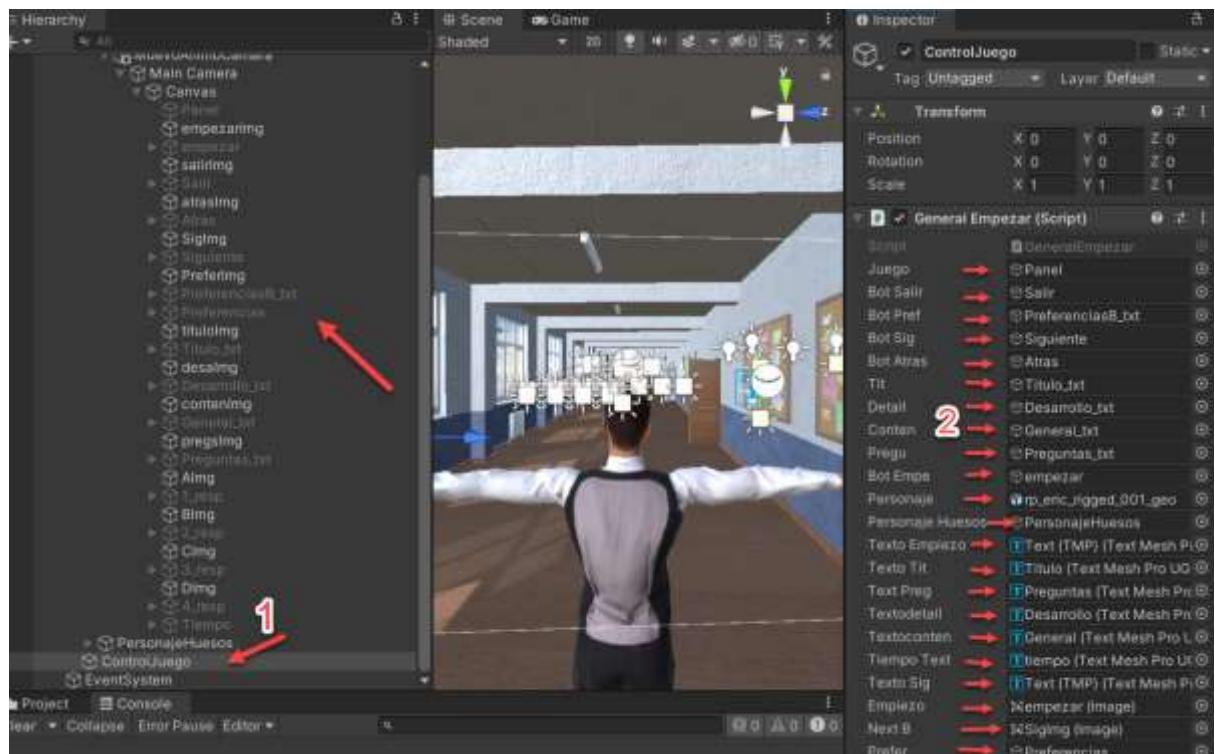


Fig 80 "Relacionamos o arrastramos cada objeto con la variable correspondiente"

```
//Se declaran las variables publicas y las asociamos con los gameobjects correspondientes, dichas variables  
//se utilizaran para dar animaciones a objetos del escenario.  
public GameObject prefer, fondoPC, papeles, papeles1, calcu, mouse;
```

Fig 81 "Fragmento de código de declaración de variables"



Fig 82 "Relacionamos o arrastramos cada objeto con la variable correspondiente"

```
//Se declara la variable estatica booleana para controlar cuando la barra de preferencias este activada o desactivada.  
public static bool prefbool = false;  
//Se declara la variable de tipo string, en la cual cargamos las preguntas con sus respectivas opciones de respuesta.  
//Las preguntas son separadas mediante “;” y las opciones de respuesta las separamos mediante “||”  
string documento = "Por favor seleccione la opción que considere correcta para seleccionar al personal que se liquidará.;1.- Análisis de ev  
//Variable en la que se cargara una a una las líneas de texto que estan separando así controlar que no este vacia.  
string linea;  
// array en donde se guardaran los tiempos de respuesta utilizados por el participante.  
string[] tiemposResp;  
// la variable pos se utiliza como contador de las actividades y utilizamos la variable contRes como contador de las respuestas correctas.  
int pos = 0, contRes = 0;  
// Se declara las siguientes listas de string en donde:  
//contenPreg, alternativaPreg, contenPregFinal, alternativaPregFinal  
/  
list<string> contenPreg, alternativaPreg, contenPregFinal, alternativaPregFinal;
```

Fig 83 "Fragmento de código de declaración de variables"

```

//Se declara un array de gameobject en donde se cargarán los gameobject que contienen las alternativas a elegir por el participante.
public GameObject[] altern;
//Se declara un array de tipo texto el cual contiene el texto correspondiente a cada alternativa a elegir.
public TextMeshProUGUI[] textalter;

```

Fig 84 "Fragmento de código de declaración de variables"

El array *altern* hace referencia a los botones que contienen las alternativas de respuestas, el array *textalter* corresponde a espacio en donde se cargará el texto de las alternativas; ya que cada pregunta cuenta con cuatro alternativas se define un tamaño de 4.



Fig 85 "Relacionamos los objetos que contienen las alternativas de la pregunta"

```

//Se declara array de string en donde cargamos las respuestas correctas.
public string[] rescor;

```

Fig 86 "Fragmento de código de declaración de variables"

A continuación, le damos un tamaño de 2 al array ya en este caso son dos preguntas, especificamos las respuestas correctas de cada pregunta, en este caso las respuestas correctas de las preguntas del ejercitario Información difícil son la respuesta 1.



Fig 87 "Se registra las respuestas correctas de las actividades"

```

//variable que utilizamos para mostrar texto en pantalla.
string retFin;
int puntajeFinal = 0;
private float StartTime;
//variable es utilizada para controlar en que panel o menú nos encontramos.
public static string menu = "Main";

```

Fig 88 "Fragmento de código de declaración de variables"

Para la navegación mediante el teclado se crearon arreglos seleccionables, arreglos de imágenes y de audios, por tanto, es importante recalcar que el orden en que se cargue los objetos en cada arreglo será el orden que se seleccione al presionar el teclado, es decir si al presionar el teclado se activa el botón en la posición 0 entonces también se activará la imagen y sonido en la posición 0.

```

/* Array de objetos seleccionables, cada objeto que se va seleccionar con el teclado.
// nextFieldMain: botones panel principal
// nextFieldPrefer: botones panel de preferencias
// nextFieldDiag: botones panel de instrucciones
// nextFieldPreg: botones panel de preguntas
// nextFieldFinal: botones panel de puntuación
// nextFieldFin: botones panel de retroalimentación
*/
public Selectable[] nextFieldMain, nextFieldPrefer, nextFieldDiag, nextFieldPreg, nextFieldFinal, nextFieldFin;
/* Array de imágenes correspondientes a cada botón del juego.
// main: imágenes panel principal
// prefImg: imágenes panel de preferencias
// dial: imágenes panel de instrucciones
// juegI: imágenes panel de preguntas
// finI: imágenes panel de retroalimentación
// alternativa: imágenes de las alternativas a elegir
// finalI: imágenes panel de puntuación */
public Image[] main, prefImg, dial, juegI, finI, alternativa, finalI;

```

Fig 89 "Fragmento de código de declaración de variables"

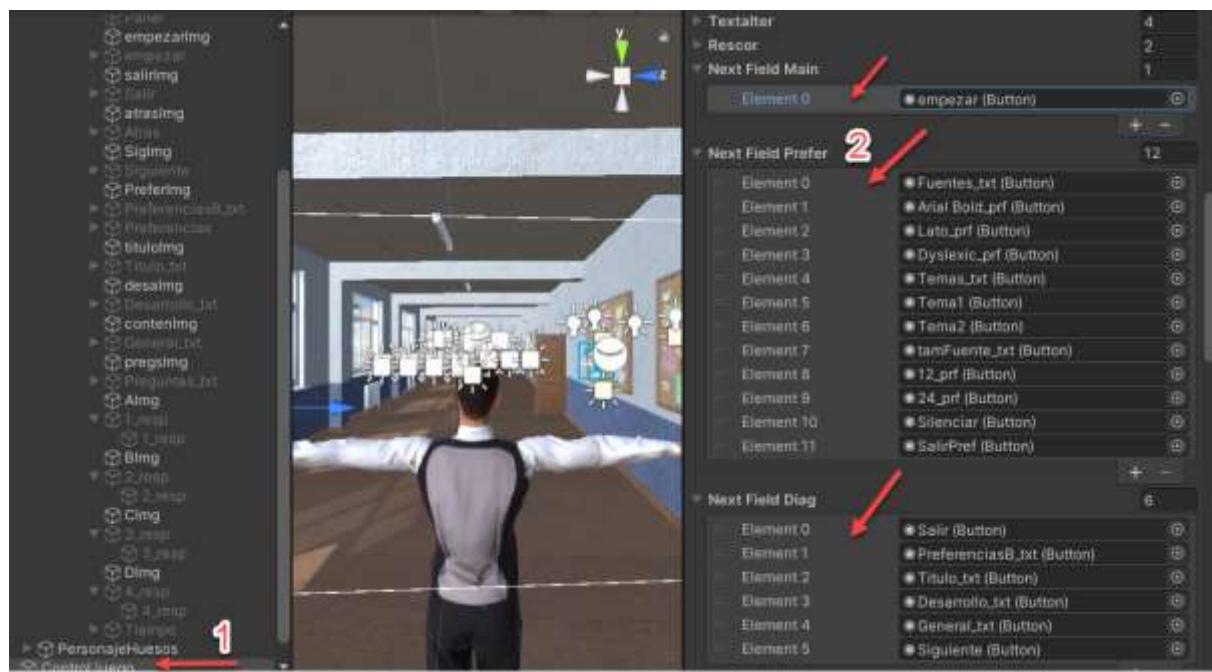


Fig 90 "Relacionamos los objetos con la variable correspondiente"

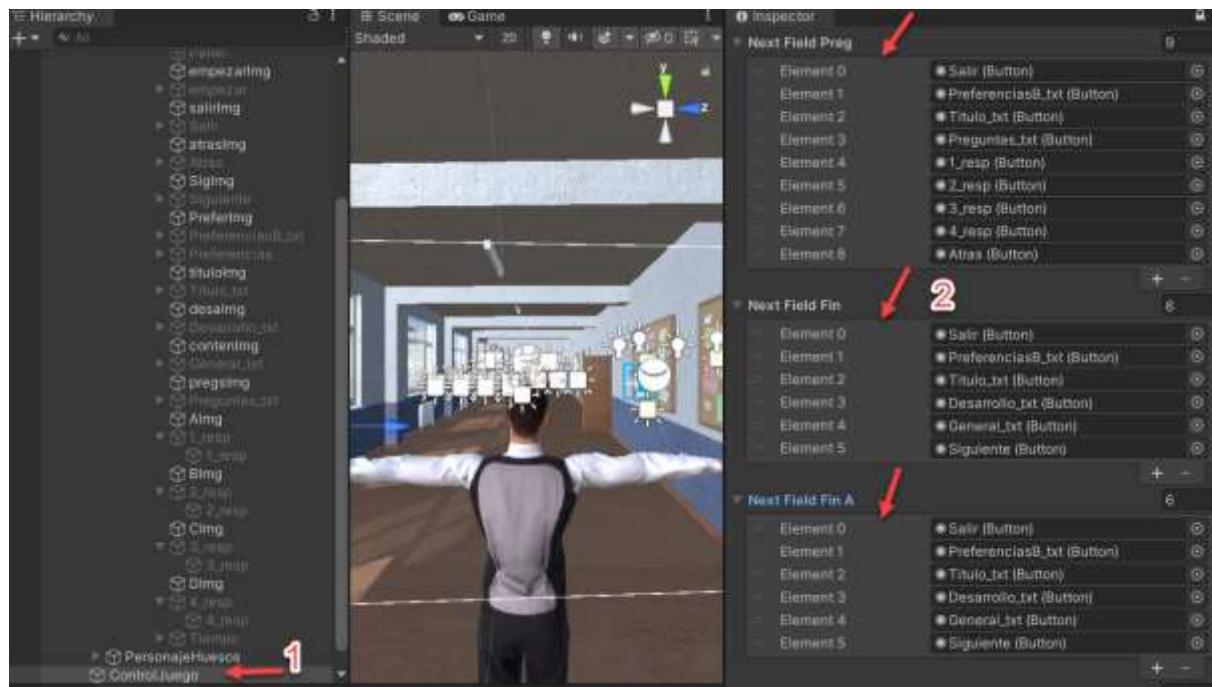


Fig 91 "Relacionamos los objetos con la variable correspondiente"

```

/*
 * Array de imágenes correspondientes a cada botón del juego.
 */
public Image[] main, prefIog, dial, juegI, finI, alternativa, finAT;

```

Fig 92 "Fragmento de código de declaración de variables"

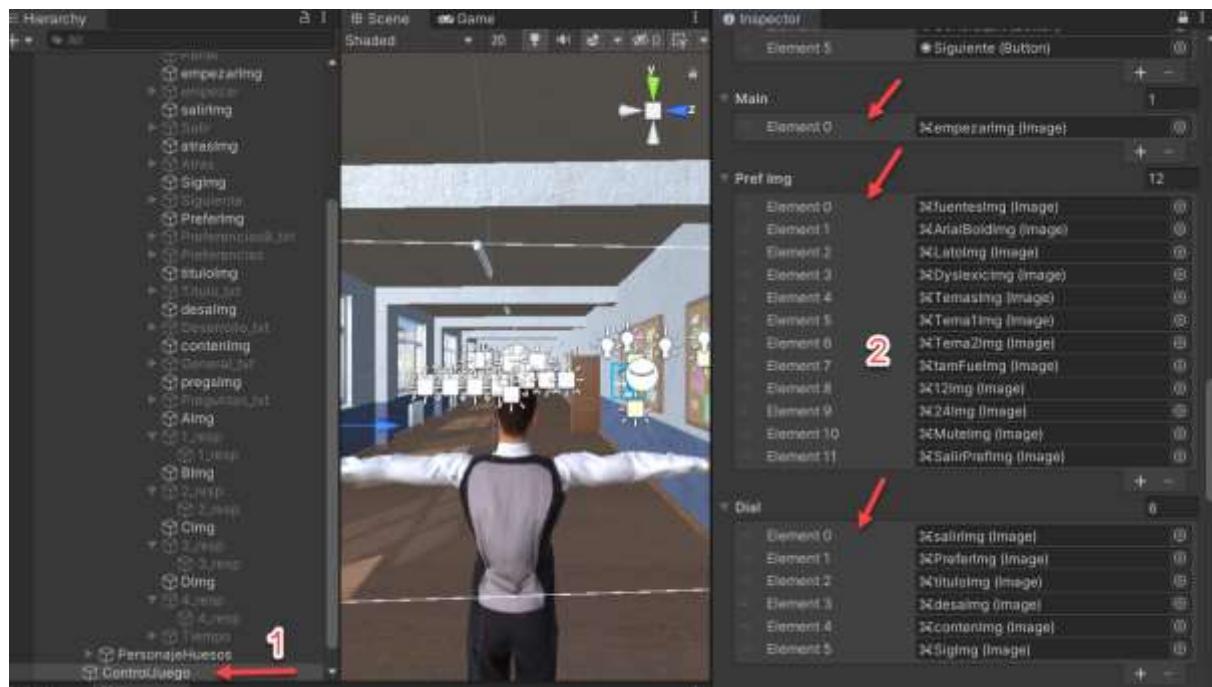


Fig 93 "Relacionamos los objetos con la variable correspondiente"

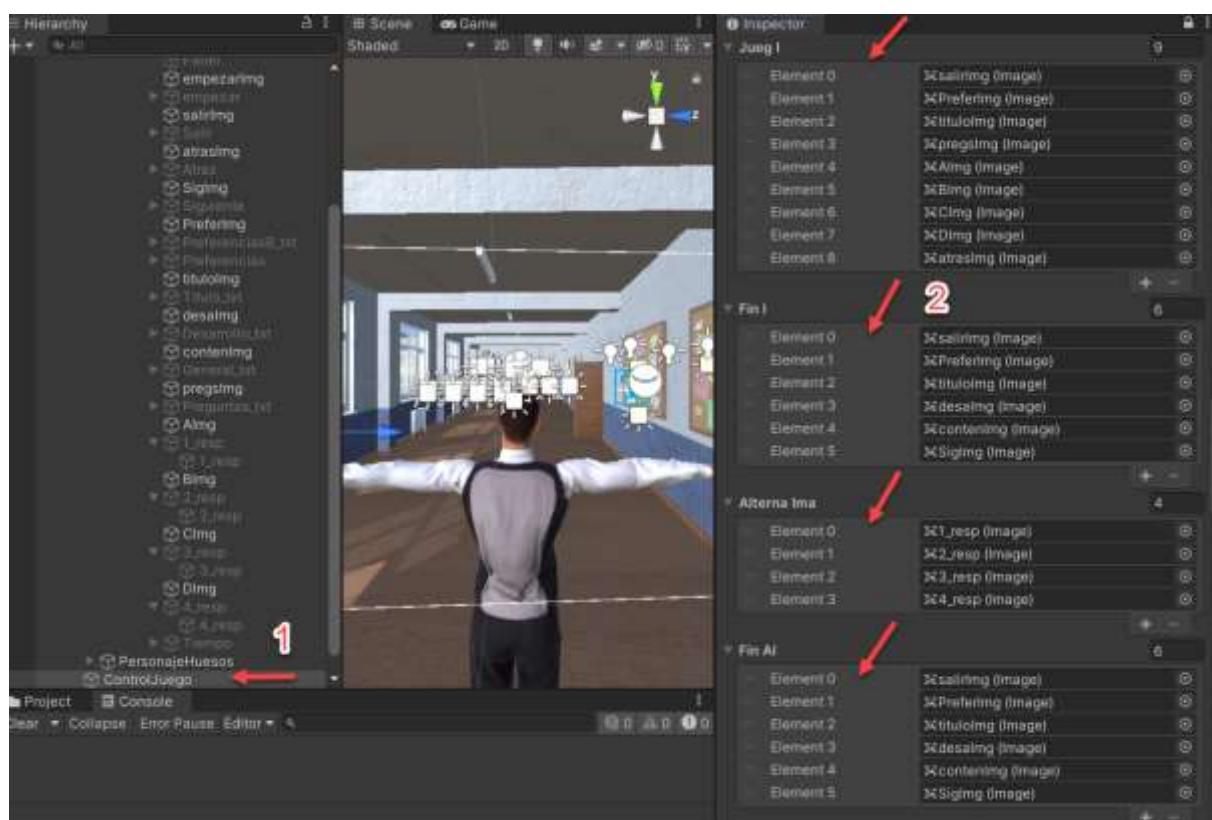


Fig 94 "Relacionamos los objetos con la variable correspondiente"

Es importante resaltar que el arreglo de objetos seleccionables y el arreglo de imágenes son los que van a funcionar con el teclado, por ejemplo, si al presionar el teclado

nos posicionamos en la tecla salir se activará también la imagen como se muestra a continuación.



Fig 95 "Ejemplo de selección de botón salir acompañado de una imagen que lo resalta"

A continuación, se declaran contadores para recorrer los arreglos de botones, audios, imágenes utilizadas para la navegación por teclado.

```

/* numC: controlador para saber en que botón del panel de preferencias estamos
numT: controlador para saber en que IMAGEN estamos */
int numC = 0, numT = 0;
/* numP: controlador para saber en que botón del panel de preferencias estamos
numPP: controlador para saber en que imágenes del panel de preferencias estamos */
int numPP = 0, numPF = 0;
/* numF: controlador para saber en que pregunta estamos
numFP: controlador para saber en que imagen de la pregunta estamos */
int numFP = 0, numFPP = 0;
/* numR: controlador para saber en que botón del panel de puntificación estamos
numRF: controlador para saber en que imágenes del panel de puntificación estamos */
int numRF = 0, numRFF = 0;
/* numRA: controlador para saber en que botón del panel de retroalimentación estamos
numRFA: controlador para saber en que imágenes del panel de retroalimentación estamos */
int numRFA = 0, numRFFA = 0;
/* variantes para recorrer en que pregunta estamos
// ap1: alternativas de la pregunta 1
// ap2: alternativas de la pregunta 2 */
int ap1 = 0, ap2 = 0;
/* variable que contiene en que número de pregunta estamos
int numPreg

```

Fig 96 "Fragmento de código de declaración de variables"

Se declara una variable de tipo AudioSource , esta será quien reproduzca todos los audios.

```

// variable de tipo AudioSource que representa al sonido principal, que es donde va a salir todos los sonidos al interactuar con el recinto
public AudioSource sonidos;

```

Fig 97 "Fragmento de código de declaración de variables"

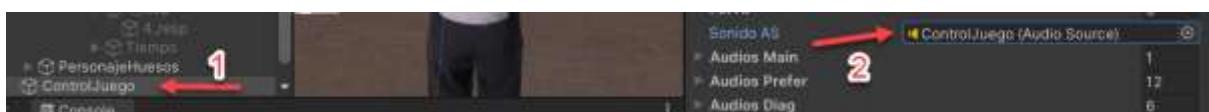


Fig 98 "Relacionamos el objeto con la variable correspondiente"

```
array de audios, con uno de estos audio se va a cargar, etc.
```

```
public AudioSource[] audiosMain, audiosPref, audiosDiag, audiosPreg, audiosP2, audiosFin, audiosFinA;
```

Fig 99"Fragmento de código en donde se declaran varios arreglos de audios"

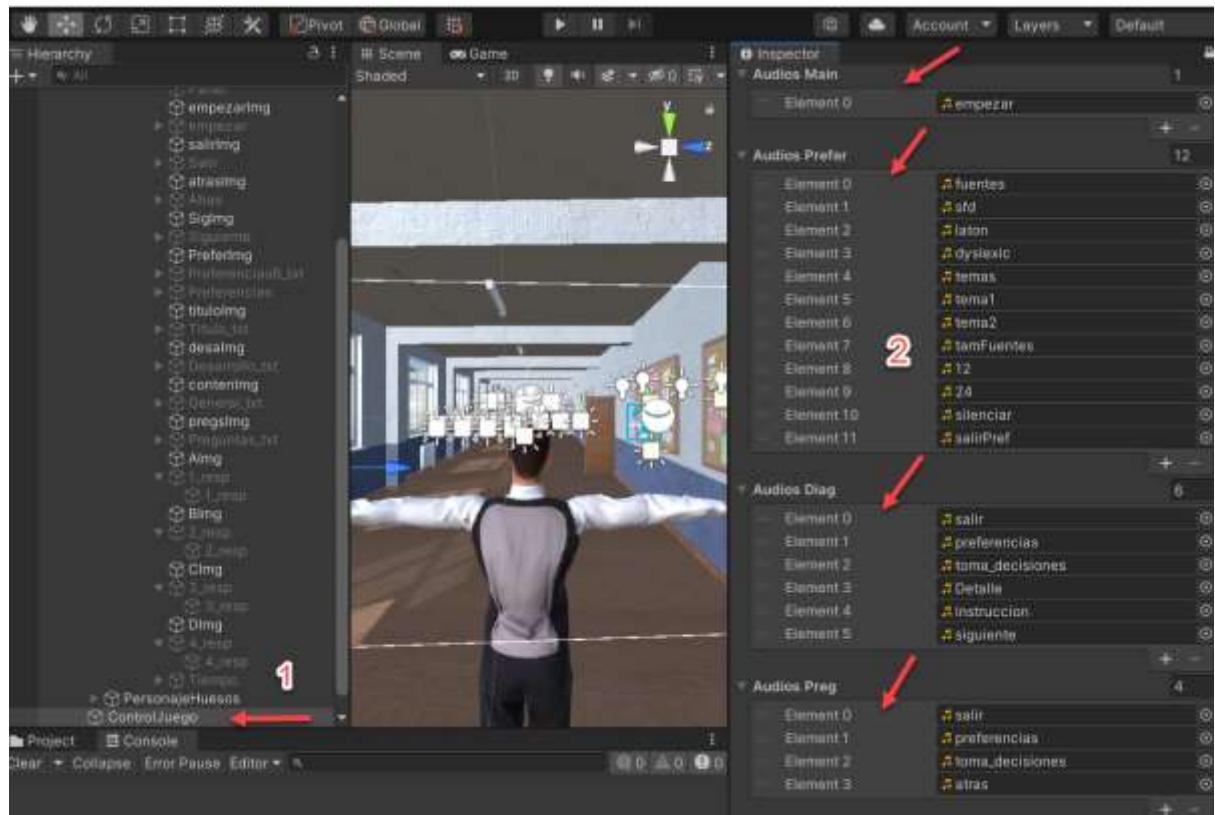


Fig 100"Relacionamos los objetos con la variable correspondiente"

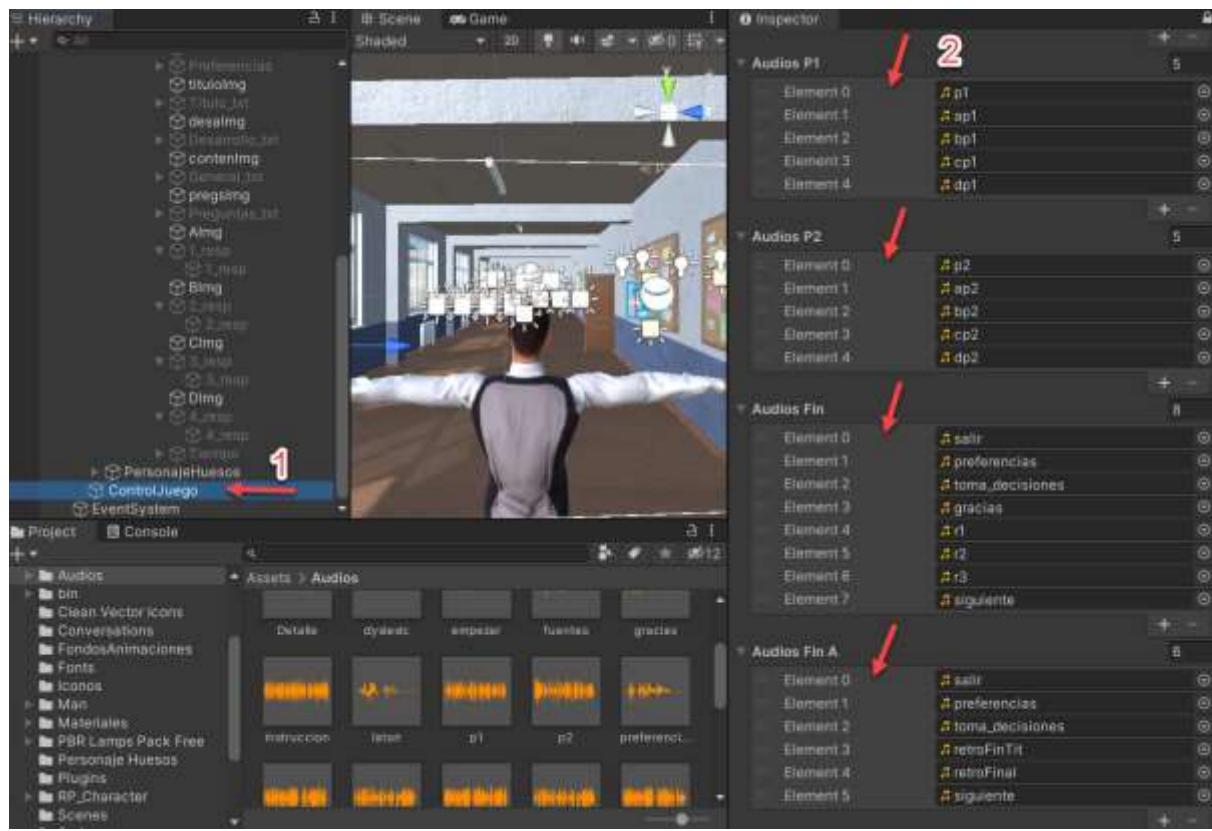


Fig 101 "Relacionamos los objetos con la variable correspondiente"

A continuación, se declaran arreglos de tipo AudioSource los cuales guardarán todos los audios a reproducir mediante el mouse.

```
public AudioSource[] audiosMainMouse, audiosPreferMouse, audiosDigMouse, audiosPregMouse, audiosFinMouse,
```

Fig 102 "Se declaran arreglos de gameobjects"

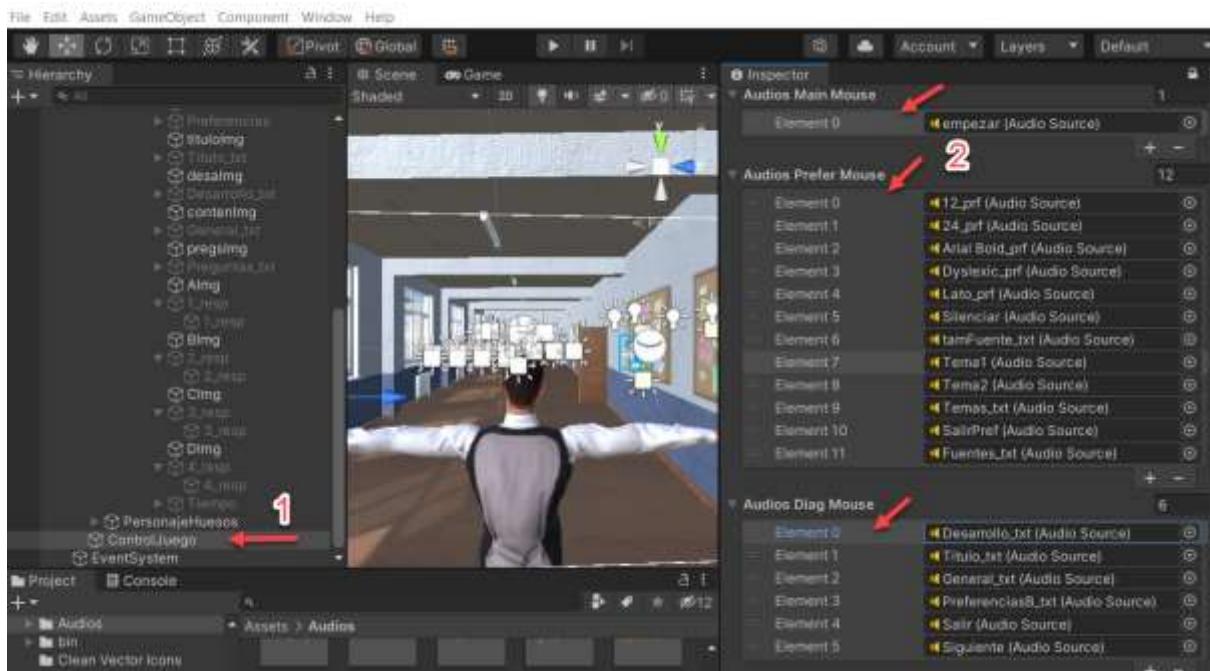


Fig 103 "Relacionamos los objetos con la variable correspondiente"

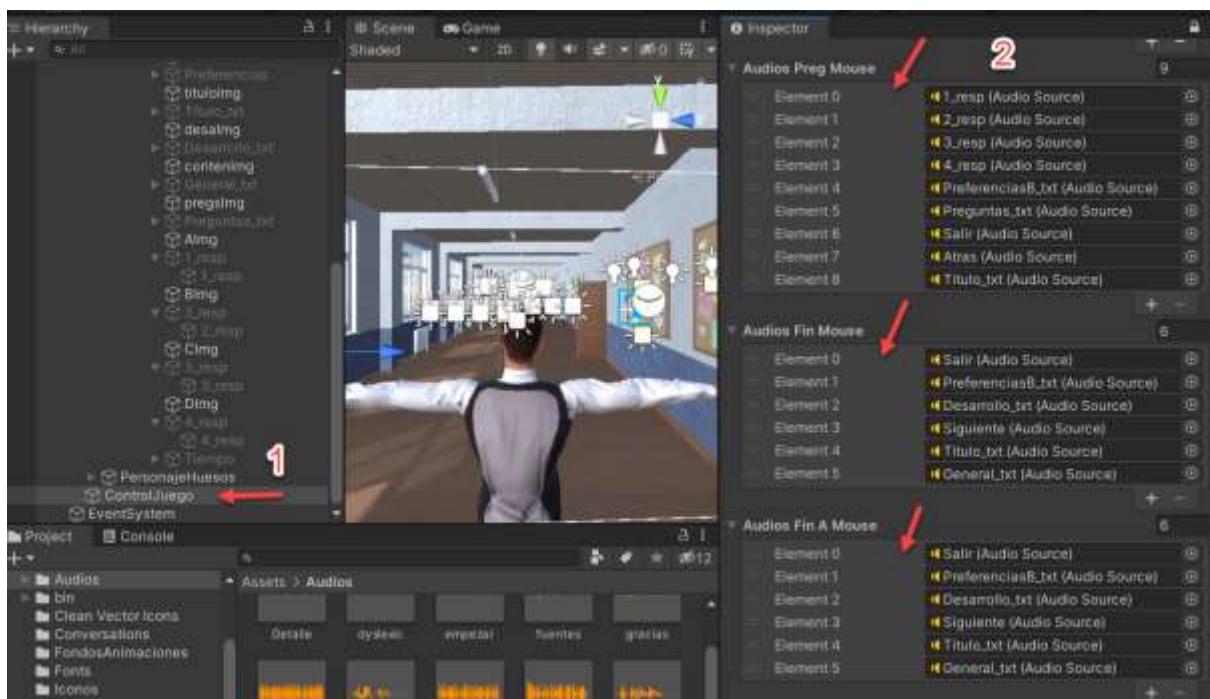


Fig 104 "Relacionamos los objetos con la variable correspondiente"

```

/* variable que será true o false cuando se mueva la cámara */
public static bool move = true;
/* variable cámara de tipo gameobject a la cual se le relaciona la cámara principal de la escena.*/
public GameObject camara;
//Lista de tipo entero en donde se guardará los números aleatorios
List<int> numerosGuardados;
/* lista en donde se guardan las preguntas. */
List<int> numPares;
/* lista de enteros, en donde se guardan las alternativas. */
List<int> listaFinalPregs;
// variable que se usa de contador de las preguntas.
int contadorPR;
/* variable que obtiene la fecha actual. */
string datetime;

```

Fig 105 "Fragmento de código de declaración de variables"

```

// botón que se usa para indicar que se ha llegado al final del juego, a este variable la asociamos al botón siguiente.
public Button sigueFin;
/* variables de tipo sprite
   // ImagenCasa: imágenes para reiniciar juego
   // ImagenSiguiiente: imágenes para botón siguiente
   */
public Sprite imagenCasa, ImagenSiguiiente;

```

Fig 106 "Fragmento de código de declaración de variables"

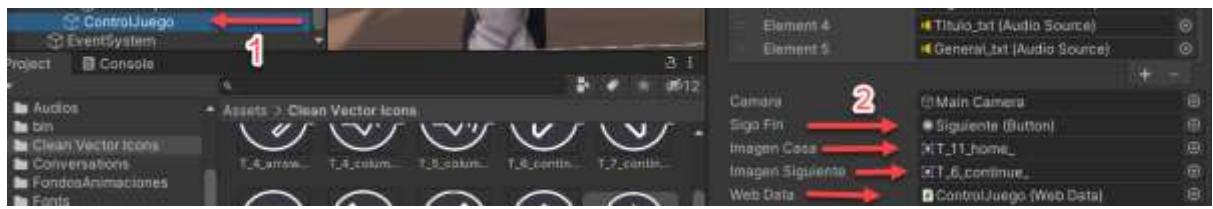


Fig 107 "Relacionamos los objetos con la variable correspondiente"

```

/* declaro el array de preguntas. */
string arrayPreguntas;
/* creación de una variable de la clase WebData,
   a la cual se pasa los campos de datos para consumir los servicios web,
   luego relacionamos esta variable con el gameObject que contiene el script WebData.cs. */
public WebData webData;
/* instancia de la clase SimuladorData */
public SimuladorData eje = new SimuladorData();
/* variable de la clase Pregunta, esta variable irá agregando dentro del JSON una nueva pregunta*/
public Pregunta ppre;
// variable que es utilizada para controlar el estado de las preguntas.
string TextoAgregoModif = "";

```

Fig 108 "Fragmento de código de declaración de variables"

### 1.2.1.2 Declaración de funciones

A continuación, se muestra el código fuente utilizado con su respectiva documentación.

```

void Start()
{
    nuevoJuego(); //Llamamos a la función para que antes que nada los valores empiecen en 0.
}

```

Fig 109 "Fragmento de código de la función Start"

```

void Update()
{
    //si menu es igual a Preguntas
    if ( GeneralEmpezar.menu == "Preguntas" )
    {
        correTiempo(); //empieza a tomar el tiempo
    }
    //si el botón empezar esta activo y el movimiento de la cámara esta activo
    if (botEmpe.activeSelf == true && move == true)
    {
        //podremos mover la vista de la cámara mediante el mouse/
        float pointer_x = Input.GetAxis("Mouse X");
        float pointer_y = Input.GetAxis("Mouse Y");
        camara.transform.Rotate(0, pointer_x * 1f, 0);
    }
    if (botEmpe.activeSelf == true) //si el botón empezar esta activo
    {
        //se desactiva los personajes/
        personaje.SetActive(false);
        personajeHuesos.SetActive(false);
    }
    Tabulando(); //llamamos a la función tabulando, para que siempre este escuchando a los eventos del teclado
}

```

Fig 110 "Fragmento de código de la función Update() en donde se implementa el movimiento de la cámara con el mouse"

```

//esta función permite salir del simulador
public void Salir()
{
    Application.Quit();
}

```

Fig 111 "Fragmento de código de la función Salir"



Fig 112 "Agregamos el evento On Click y llamamos a la función Salir"

La función `nuevoJuego()` permite al estudiante volver a la interfaz principal en donde podrá iniciar de nuevo la interacción con el simulador laboral, inicializando todos los valores de las variables en cero.

```
109     public void nuevoJuego()
110     {
111         // se setea el tiempo de inicio para cero
112         eje.setTiempoInit((System.DateTime.Now.Hour.ToString("00") + ":" + System.DateTime.Now.Minute.ToString("00") + ":" + System.DateTime.Now.Second.ToString("00")));
113         tiempoResp = new string[2];
114         arrayPreguntas = new string[10]; // cantidad de preguntas
115         eje.ReinicioPreguntas(); //limpieza este array para que inicie en ceros las variables
116         try
117         {
118             contenidoPreg = new List<string>(); //crea una lista para
119             linea = documento;
120             if (linea != null) // si no hay linea no hace nada
121             {
122                 //guarda las preguntas las cuales vienen del archivo .txt
123                 contenidoPregFinal = new List<string>(linea.Split(new string[] { "\r\n" }, StringSplitOptions.None));
124             }
125         }
126         catch (Exception e)
127         {
128             Debug.Log(e);
129         }
130     }
```

Fig 113 "Fragmento de código de la función nuevoJuego"

La función *Tabulando ()* principalmente se hace cargo de la navegación mediante el teclado en donde para navegar se ha definido las teclas: navegación hacia atrás(alt, fecha izquierda, flecha arriba) y para navegar hacia adelante(tab, flecha derecha y flecha abajo).

```
92     public void Tabulando()
93     {
94         if (GeneralEmpezar.menu == "Main")//siempre empieza con el panel main
95         {
96             //introducción al panel main
97             if (botEmp.activeSelf == true)//comprobamos que estamos en el panel principal si el botón empezar está activo
98             {
99                 //mando a seleccionar el objeto dentro del arreglo nextFieldMain que se encuentra en esa posición
100                 nextFieldMain[nextFieldMain.Length - 1].Select();
101
102                 /* Si se presiona alguna de estas teclas, se irá seleccionando hacia adelante */
103                 if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
104                 {
105                     /* cuando se detener los audios del mouse cuando detecte una entrada por teclado,
106                     y de esa manera no se oigan los audios que están reproduciéndose */
107                     foreach (var item in audiosMainMouse)
108                     {
109                         item.Stop();
110                     }
111                     sonidoAS.clip = audiosMain[0]; //asigna el audio que se encuentra en la posición 0
112                     sonidoAS.Play(); //reproducir el audio en esa posición
113
114                     /* Y si se presiona alguna de estas teclas se irá seleccionando hacia atrás*/
115                     else if (Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow) || Input.GetKeyDown(KeyCode.LeftAlt))
116                     {
117                         foreach (var item in audiosMainMouse) //mando a desactivar todos los audios del mouse que estén reproduciéndose
118                         {
119                             item.Stop();
120                         }
121                         sonidoAS.clip = audiosMain[0];
122                         sonidoAS.Play();
123                     }
124                 }
125             }
126         }
127     }
```

Fig 114 "Fragmento de código de la función Tabulando parte 1"

```

    case 17 (GeneralEmpiezar.menu == "Juegos")
    {
        //caso de empiezo
        if (empiezo.enabled == false){//si el siguiente comando comienza a cuando el botón empieza este desactivado
            I //se activa el comando que hace lo mismo del botón inicio
            foreach (var item in menu)
            {
                //comprueba si el comando que se ejecuta es el que se ha pulsado
                if (item.enabled == true)
                {
                    if (numC < nextFieldDiag.Length - 1) //si el comando menu es menor al tamaño del arreglo nextFieldDiag - 1
                    {
                        //se ejecuta el comando que hace lo mismo del botón que se ha pulsado
                        if (nextFieldDiag[numC].Select() == true) //si el comando menu es igual al comando que se ha pulsado
                        {
                            //se ejecuta el comando que hace lo mismo del botón que se ha pulsado
                            dial(menu).enabled = true;
                        }
                    }
                    else //si el comando menu es mayor al tamaño del arreglo nextFieldDiag que se encuentra en la posición actual
                    {
                        nextFieldDiag[numC].Select();
                        //se ejecuta el comando que hace lo mismo del botón que se ha pulsado
                        dial(menu).enabled = true;
                    }
                }
            }
        }
        else if (numC > 0)
        {
            //se ejecuta el comando que hace lo mismo del botón que se ha pulsado
            numC = nextFieldDiag.Length - 2;
            numI = nextFieldDiag.Length - 1;
            nextFieldDiag[numC].Select();
            dial(menu).enabled = true;
        }
        if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
        {
            //se ejecuta el comando que hace lo mismo del botón que se ha pulsado
            if (numC < nextFieldDiag.Length - 1) //si el comando menu es menor al tamaño del arreglo nextFieldDiag - 1
            {
                //se ejecuta el comando que hace lo mismo del botón que se ha pulsado
                if (nextFieldDiag[numC].Select() == true) //si el comando menu es igual al comando que se ha pulsado
                {
                    //se ejecuta el comando que hace lo mismo del botón que se ha pulsado
                    sonidoAS.Stop();
                    sonidoAS.clip = audioDiag[4];
                    sonidoAS.Play();
                    sonidoAS.loop = false;
                    dial(menu).enabled = false;
                }
            }
            numC++; //se aumenta en 1 la posición de numC
            numI--;
            //se ejecuta el comando que hace lo mismo del botón que se ha pulsado
            nextFieldDiag[numC].Select();
            //se ejecuta el comando que hace lo mismo del botón que se ha pulsado
            sonidoAS.Stop();
            sonidoAS.clip = audioDiag[4];
            sonidoAS.Play();
            sonidoAS.loop = false;
            dial(menu).enabled = false;
            dial(numI).enabled = true;
        }
    }
}

```

*Fig 115"Fragmento de código de la función Tabulando parte 2"*

```

    else if (Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow) || Input.GetKeyDown(KeyCode.LeftAlt))
        /* cuando se detiene los aviones del mouse cuando detecta una entrada por teclado,
        y en ese momento ya se crean los aviones con estos reproduciendose*/
        foreach (var item in audiosDiagMouse)
    {
        item.Stop(); //detenga tanto los aviones del mouse
    }
    //si el contador numC es igual a 8
    if (numC == 8)
    {
        numC = nextFieldDiag.Length - 1; //invierte el contador numC si es decir seleccionando el ultimo elemento del array.
        numI = nextFieldDiag.Length - 1; //invierte el contador numI en 1 es decir seleccionando el ultimo elemento del array.
        nextFieldDiag[numC].Select(); //selecciona el boton en la nueva posicion.
        sonidoAS.Stop(); //detenga tanto los sonidos.
        sonidoAS.Clip = audiosDiag[numC]; //carga el audio en la nueva posicion.
        sonidoAS.Play(); //reproduce el sonido.
        dial[0].enabled = false; //desactiva la imagen del boton de la posicion 0.
        dial[numI].enabled = true; //activa la imagen del boton de la posicion actual.
    }
    //si el contador es menor al tamaño del array
    else if (numC < nextFieldDiag.Length - 1)
    {
        numC++; //incrementa en 1 a la posicion del contador.
        numI++; //incrementa en 1 a la posicion del contador.
        nextFieldDiag[numC].Select(); //invierte y selecciona el boton en la nueva posicion.
        sonidoAS.Stop(); //detenga tanto los sonidos.
        sonidoAS.Clip = audiosDiag[numC]; //carga el audio en la nueva posicion.
        sonidoAS.Play(); //reproduce el sonido.
        dial[numI - 1].enabled = false; //desactiva la imagen del boton de la posicion actual mas 1.
        dial[numI].enabled = true; //activa la imagen del boton en la posicion actual.
    }
}
// si el tamaño del contador numC es igual a 8, es decir del array nextFieldDiag
else if (numC == nextFieldDiag.Length - 1)
{
    /* si el se presiona alguna de estas teclas */
    if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
        /* cuando se detiene los aviones del mouse cuando detecta una entrada por teclado,
        y en ese momento ya se crean los aviones que estan reproduciendose*/
        foreach (var item in audiosDiagMouse)
    {
        item.Stop(); //detenga tanto los aviones
    }
    dial[numI].enabled = false; //desactiva el boton que esta en esa posicion.
    numC = 0; //se inicializa numC en 0
    numI = 0; //se inicializa numI en 0
    nextFieldDiag[numC].Select(); //selecciona el boton en la posicion actual.
    sonidoAS.Stop(); //detenga tanto los sonidos.
    sonidoAS.Clip = audiosDiag[numC]; //carga el audio en la nueva posicion.
    sonidoAS.Play(); //reproduce el sonido.
    dial[numI].enabled = true; //activa la imagen del boton en la posicion actual.
}

```

Fig 116 "Fragmento de código de la función Tabulando parte 3"

```

337     /* Y si se presiona alguna de estas teclas */
338     else if (Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow) || Input.GetKeyDown(KeyCode.LeftAlt))
339     {
340         numC--;//decremento en 1 a la posición del contador.
341         numI--;//decremento en 1 a la posición del contador.
342
343         /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
344         y de esa manera no se cruzan los audios que estén reproduciéndose*/
345         foreach (var item in audiosDiagMouse)
346         {
347             item.Stop();//detenga todos los audios.
348         }
349         dial[numI + 1].enabled = false;//desactivó la imagen del botón de la posición actual más 1.
350         nextFieldDiag[numC].Select();//selección el botón en la posición actual.
351         sonidoAS.Stop();//detengo todos los audios que estén reproduciéndose,
352         sonidoAS.clip = audiosDiag[numC];//cargo el audio en la nueva posición,
353         sonidoAS.Play();//mando a reproducir el audio,
354         dial[numI].enabled = true;//activó la imagen del botón de la posición actual.
355     }
356 }
357 //si el botón siguiente está desactivado significa que llegó al panel de preguntas
358 if (botSig.activeSelf == false)
359 {
360     //al método MenuSel el cual se utiliza para controlar en que menu nos encontramos le paso el nombre Preguntas.
361     GeneralEmpezar.MenuSel("Preguntas");
362     // mando a desactivar todos los objetos del panel de instrucciones/
363     foreach (var item in dial)
364     {
365         item.enabled = false;//desactivó todos los objetos.
366     }
367     numC = 0;//inicializo el contador en 0.
368     numI = 0;//inicializo el contador en 0.
369 }
370
371 //fin condicional panel de instrucciones
372
373 //si menu es igual a preguntas
374 else if (GeneralEmpezar.menu == "Preguntas")
375 {
376     //inicio condicional panel de preguntas
377     //si el botón siguiente está desactivado
378     if (botSig.activeSelf == false)
379     {
380         nextB.enabled = false;//desactivó la imagen del botón siguiente.
381
382         //Si el contador numPP es menor al tamaño del arreglo nextFieldPreg.
383         if (numPP < nextFieldPreg.Length - 1)
384         {
385             //Y si el contador es igual a 0
386             if (numPP == 0)
387             {
388                 nextFieldPreg[numPP].Select();//selecciona el botón en la posición actual.
389                 juegI[numPP].enabled = true;//activa la imagen del botón en la posición actual.
390             }
391         }
392     }
393 }

```

Fig 117"Fragmento de código de la función Tabulando parte 4"

```

if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
{
    // numero de respuesta que se escoge del mouse cuando detecta una tecla por teclado.
    // o de los botones de los cuales los botones que estan en el orden de la pregunta
    Turnear();
}

// item.Stop();/detenga todos los sonidos del mouse.
if (numPP == 0) // si el contador es igual a 0
{
    sonidoAS.Stop(); // detenga todos los sonidos
    sonidoAS.clip = audiosPreg[numPP]; // carga el audio en la nueva posición.
    sonidoAS.Play(); // comienza a reproducir el audio.
}

numPP++; // aumenta en 1 la la posición de mi botón.
numPP+=1; // aumenta en 1 a la posición en la imágenes del botón.

nextFieldPreg[numPP].Select(); // selecciona el botón en la posición actual.
sonidoAS.Stop(); // detenga todos los sonidos.
if (numPP == 2) // si el contador de preguntas numPP es menor o igual a 2.
{
    sonidoAS.clip = audiosPreg[numPP]; // carga el audio en la nueva posición.
    sonidoAS.Play(); // comienza a reproducir el audio.
}

// si el contador numPP es menor o igual a 2 donde el anidado si menor o igual a 7(és la ultima alternativa en la pregunta)
else if (numPP > 2 && numPP < 7)
{
    // si el número del botón es igual a Preguntas_txt
    if (nextFieldPreg[numPP].name == "Preguntas_txt")
    {
        // si el texto de la pregunta es igual a
        if (textPreg.text == "Por favor seleccione la opción que considere correcta para seleccionar al personal que se liquidará:")
        {
            sonidoAS.clip = audiosP1[0]; // carga el audio de la pregunta 1 en la posición 0.
            sonidoAS.Play(); // comienza a reproducir el audio.
        }
    }
    // si el texto de la pregunta es igual a
    else if (textPreg.text == "Ahora elija el discurso que usted utilizaría para comunicar a su compañero:")
    {
        sonidoAS.clip = audiosP2[0]; // carga el audio de la pregunta 2 en la posición 0.
        sonidoAS.Play(); // comienza a reproducir el audio.
    }
}
}

// si el número del botón es igual a 1_resp
if (nextFieldPreg[numPP].name == "1_resp")
{
    ap1[1]; // asigna el valor de 1 a ap1.
    sonidoAS.clip = audiosP1[ap1]; // carga el audio de la posición actual.
    sonidoAS.Play();
}

// si el número del botón es igual a 2_resp
else if (nextFieldPreg[numPP].name == "2_resp")
{
    ap1[2]; // asigna el valor de 2 a ap1.
    sonidoAS.clip = audiosP1[ap1];
    sonidoAS.Play();
}

// si el número del botón es igual a 3_resp
else if (nextFieldPreg[numPP].name == "3_resp")
{
    ap1[3]; // asigna el valor de 3 a ap1.
    sonidoAS.clip = audiosP1[ap1]; // carga el audio de la posición actual.
    sonidoAS.Play();
}

// si el número del botón es igual a 4_resp
else if (nextFieldPreg[numPP].name == "4_resp")
{
    ap1[4]; // asigna el valor de 4 a ap1.
    sonidoAS.clip = audiosP1[ap1]; // carga el audio de la posición actual.
    sonidoAS.Play();
}

// si el número del botón es igual a 5_resp
else if (nextFieldPreg[numPP].name == "5_resp")
{
    ap1[5]; // asigna el valor de 5 a ap1.
    sonidoAS.clip = audiosP1[ap1]; // carga el audio de la posición actual.
    sonidoAS.Play();
}

// si el número del botón es igual a 6_resp
else if (nextFieldPreg[numPP].name == "6_resp")
{
    ap1[6]; // asigna el valor de 6 a ap1.
    sonidoAS.clip = audiosP1[ap1]; // carga el audio de la posición actual.
    sonidoAS.Play();
}

// si el número del botón es igual a 7_resp
else if (nextFieldPreg[numPP].name == "7_resp")
{
    ap1[7]; // asigna el valor de 7 a ap1.
    sonidoAS.clip = audiosP1[ap1]; // carga el audio de la posición actual.
    sonidoAS.Play();
}

// si el número del botón es igual a Apunt
else if (textPreg.text == "Ahora elija el discurso que usted utilizaría para comunicar a su compañero:")
{
    ap1[8]; // asigna el valor de 8 a ap1.
    sonidoAS.clip = audiosP2[ap1]; // carga el audio de la posición actual.
    sonidoAS.Play();
}

// si el contador es menor a 7
if (numPP < 7)
{
    // carguó audio/escoge el audio de la posición actual - 1.
    sonidoAS.clip = audiosPreg[numPP - 1];
    sonidoAS.Play(); // comienza a reproducir el audio.
}

juegE[numPP - 1].enabled = false; // desactiva la imagen
juegE[numPP].enabled = true; // activa la imagen de la posición actual.

```

Fig 118"Fragmento de código de la función Tabulando parte 5"

```

if (textPreg.text == "Por favor seleccione la opción que considere correcta para seleccionar al personal que se liquidará:")
{
    // si el número del botón es igual a 1_resp
    if (nextFieldPreg[numPP].name == "1_resp")
    {
        ap1[1]; // asigna el valor de 1 a ap1.
        sonidoAS.clip = audiosP1[ap1]; // carga el audio de la posición actual.
        sonidoAS.Play();
    }

    // si el número del botón es igual a 2_resp
    else if (nextFieldPreg[numPP].name == "2_resp")
    {
        ap1[2]; // asigna el valor de 2 a ap1.
        sonidoAS.clip = audiosP1[ap1];
        sonidoAS.Play();
    }

    // si el número del botón es igual a 3_resp
    else if (nextFieldPreg[numPP].name == "3_resp")
    {
        ap1[3]; // asigna el valor de 3 a ap1.
        sonidoAS.clip = audiosP1[ap1]; // carga el audio de la posición actual.
        sonidoAS.Play();
    }

    // si el número del botón es igual a 4_resp
    else if (nextFieldPreg[numPP].name == "4_resp")
    {
        ap1[4]; // asigna el valor de 4 a ap1.
        sonidoAS.clip = audiosP1[ap1]; // carga el audio de la posición actual.
        sonidoAS.Play();
    }

    // si el número del botón es igual a 5_resp
    else if (nextFieldPreg[numPP].name == "5_resp")
    {
        ap1[5]; // asigna el valor de 5 a ap1.
        sonidoAS.clip = audiosP1[ap1]; // carga el audio de la posición actual.
        sonidoAS.Play();
    }

    // si el número del botón es igual a 6_resp
    else if (nextFieldPreg[numPP].name == "6_resp")
    {
        ap1[6]; // asigna el valor de 6 a ap1.
        sonidoAS.clip = audiosP1[ap1]; // carga el audio de la posición actual.
        sonidoAS.Play();
    }

    // si el número del botón es igual a 7_resp
    else if (nextFieldPreg[numPP].name == "7_resp")
    {
        ap1[7]; // asigna el valor de 7 a ap1.
        sonidoAS.clip = audiosP1[ap1]; // carga el audio de la posición actual.
        sonidoAS.Play();
    }

    // si el número del botón es igual a Apunt
    else if (textPreg.text == "Ahora elija el discurso que usted utilizaría para comunicar a su compañero:")
    {
        ap1[8]; // asigna el valor de 8 a ap1.
        sonidoAS.clip = audiosP2[ap1]; // carga el audio de la posición actual.
        sonidoAS.Play();
    }

    // si el contador es menor a 7
    if (numPP < 7)
    {
        // carguó audio/escoge el audio de la posición actual - 1.
        sonidoAS.clip = audiosPreg[numPP - 1];
        sonidoAS.Play(); // comienza a reproducir el audio.
    }

    juegE[numPP - 1].enabled = false; // desactiva la imagen
    juegE[numPP].enabled = true; // activa la imagen de la posición actual.
}

```

Fig 119"Fragmento de código de la función Tabulando parte 6"

```

515 //Si presiona las siguientes teclas
516 else if(Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt))
517 {
518     /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
519     y de esa manera no se cruzan los audios que estén reproduciéndose*/
520     foreach (var item in audiosPregMouse)
521     {
522         item.Stop(); //detengo todos los audios.
523     }
524     if (numPP == 0)//si el contador numPP es igual a 0
525     {
526         //iniciamos el contador numPP en 8 es decir seleccionamos el último elemento del arreglo.
527         numPP = nextFieldPreg.Length - 1;
528         //iniciamos el contador numPP en 8 es decir seleccionamos el último elemento del arreglo.
529         numFPP = nextFieldPreg.Length - 1;
530         nextFieldPreg[numPP].Select(); //selecciono el botón en la posición actual.
531         sonidoAS.Stop();
532         sonidoAS.clip = audiosPreg[3]; //cargo el audio del arreglo audiosPreg de la posición 3.
533         sonidoAS.Play(); //mando a reproducir el audio,
534         juegI[0].enabled = false; //mando a desactivar la imagen de la posición 0.
535         juegI[numFPP].enabled = true; //activo la imagen de la posición actual.
536     }
537     if (numPP == 1)//si el contador es igual a 1
538     {
539         numPP--; //decremento en 1 a la posición del contador.
540         numFPP--; //decremento en 1 a la posición del contador.
541         nextFieldPreg[numPP].Select(); //selecciono el botón en la posición actual.
542         //cargo el audio del arreglo audiosPreg de la posición actual.
543         sonidoAS.clip = audiosPreg[numPP];
544         sonidoAS.Play(); //mando a reproducir el audio,
545         //mando a desactivar la imagen de la posición actual +1
546         juegI[numFPP + 1].enabled = false;
547         juegI[numFPP].enabled = true; //activo la imagen de la posición actual.
548     }
549     if (numPP == 2)//si el contador es igual a 2
550     {
551         numPP--; //decremento en 1 a la posición del contador.
552         numFPP--; //decremento en 1 a la posición del contador.
553         nextFieldPreg[numPP].Select(); //selecciono el botón en la posición actual.
554         //cargo el audio del arreglo audiosPreg de la posición actual.
555         sonidoAS.clip = audiosPreg[numPP];
556         sonidoAS.Play();
557         //mando a desactivar la imagen de la posición actual +1
558         juegI[numFPP + 1].enabled = false;
559         juegI[numFPP].enabled = true; //activo la imagen de la posición actual.
560     }
561     if (numPP == 3)//si el contador es igual a 2
562     {
563         numPP--; //decremento en 1 a la posición del contador.
564         numFPP--; //decremento en 1 a la posición del contador.
565         nextFieldPreg[numPP].Select(); //selecciono el botón en la posición actual.
566         sonidoAS.clip = audiosPreg[numPP];
567         sonidoAS.Play();
568         //mando a desactivar la imagen de la posición actual +1
569         juegI[numFPP + 1].enabled = false;
570         juegI[numFPP].enabled = true; //activo la imagen de la posición actual.
571     }
572     //si el contador es menor a 3 o menor al tamaño del arreglo

```

Fig 120 "Fragmento de código de la función Tabulando parte 7"

```

    // si el número del botón es igual a 1,2,3,4
    if (numPP > 1 && numPP < nextFieldPreg.Length - 1)
    {
        numPP++; // incremento en 1 a la posición del contenido.
        nextFieldPreg[numPP].Select(); // selecciono el botón en la posición actual.
        sonidoAS.Stop();
        // si el texto de la pregunta es igual a
        if (textPreg.text == "Por favor seleccione la opción que considere correcta para seleccionar al personal que se liquidará.")
        {
            // si el número del botón es igual a 1,2,3,4
            if(nextFieldPreg[numPP].name == "1_resp")
            {
                ap1[1];
                sonidoAS.clip = audiosP1[ap1];
                sonidoAS.Play();
            }
            // si el número del botón es igual a 2,3,4
            else if(nextFieldPreg[numPP].name == "2_resp")
            {
                ap1[2];
                sonidoAS.clip = audiosP1[ap1];
                sonidoAS.Play();
            }
            // si el número del botón es igual a 3,4
            else if(nextFieldPreg[numPP].name == "3_resp")
            {
                ap1[3];
                sonidoAS.clip = audiosP1[ap1];
                sonidoAS.Play();
            }
            // si el número del botón es igual a 4,5
            else if(nextFieldPreg[numPP].name == "4_resp")
            {
                ap1[4];
                sonidoAS.clip = audiosP1[ap1];
                sonidoAS.Play();
            }
        }
        // si el texto de la pregunta es igual a
        else if (textPreg.text == "Ahora elija el discurso que usted utilizaría para comunicar a su compañero:")
        {
            ap1[5];
            sonidoAS.clip = audiosP1[ap1];
            sonidoAS.Play();
        }
    }
}

// si el texto de la pregunta es igual a
if (nextFieldPreg[numPP].name == "Preguntas.txt")
{
    // si el texto de la pregunta es igual a
    if (textPreg.text == "Por favor seleccione la opción que considere correcta para seleccionar al personal que se liquidará.")
    {
        sonidoAS.clip = audiosP2[0];
        sonidoAS.Play();
    }
    // si el texto de la pregunta es igual a
    else if (textPreg.text == "Ahora elija el discurso que usted utilizaría para comunicar a su compañero:")
    {
        sonidoAS.clip = audiosP2[1];
        sonidoAS.Play();
    }
}

jueg[numPP - 1].enabled = false; // desactiva la imagen de la posición actual
jueg[numPP].enabled = true; // activa la imagen de la posición actual

// si el contador es igual al tamaño del array
else if (numPP == nextFieldPreg.Length - 1)
{
    // si se presiona alguna de estas teclas
    if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
    {
        // se hace e detiene los audio del mouse cuando detecta una entrada por teclado
        // o una salida de teclado los cuales que estén reproduciéndose
        foreach (var item in audiosPregMouse)
        {
            item.Stop();
        }
        jueg[numPP].enabled = false; // desactiva la imagen de la posición actual
        numPP = 0; // inicializa el contador en 0;
        numPP = 1; // actualiza el contador en 1;
        ap1 = 0; // actualiza el contenido de la pregunta en 0;
        ap2 = 0; // actualiza el contenido de la pregunta en 0;
        nextFieldPreg[numPP].Select(); // selecciono el botón en la posición actual.
        sonidoAS.Stop();
        sonidoAS.clip = audiosPreg[numPP]; // cargo el audio de la posición actual
        sonidoAS.Play();
        jueg[numPP].enabled = true; // activa la imagen de la posición actual.
    }
}

```

Fig 121 "Fragmento de código de la función Tabulando parte 8"

```

    // si el número del botón es igual a
    if (nextFieldPreg[numPP].name == "Preguntas.txt")
    {
        // si el texto de la pregunta es igual a
        if (textPreg.text == "Por favor seleccione la opción que considere correcta para seleccionar al personal que se liquidará.")
        {
            sonidoAS.clip = audiosP2[0];
            sonidoAS.Play();
        }
        // si el texto de la pregunta es igual a
        else if (textPreg.text == "Ahora elija el discurso que usted utilizaría para comunicar a su compañero:")
        {
            sonidoAS.clip = audiosP2[1];
            sonidoAS.Play();
        }
    }

    jueg[numPP - 1].enabled = false; // desactiva la imagen de la posición actual
    jueg[numPP].enabled = true; // activa la imagen de la posición actual

    // si el contador es igual al tamaño del array
    else if (numPP == nextFieldPreg.Length - 1)
    {
        // si se presiona alguna de estas teclas
        if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
        {
            // se hace e detiene los audio del mouse cuando detecta una entrada por teclado
            // o una salida de teclado los cuales que estén reproduciéndose
            foreach (var item in audiosPregMouse)
            {
                item.Stop();
            }
            jueg[numPP].enabled = false; // desactiva la imagen de la posición actual
            numPP = 0; // inicializa el contador en 0;
            numPP = 1; // actualiza el contador en 1;
            ap1 = 0; // actualiza el contenido de la pregunta en 0;
            ap2 = 0; // actualiza el contenido de la pregunta en 0;
            nextFieldPreg[numPP].Select(); // selecciono el botón en la posición actual.
            sonidoAS.Stop();
            sonidoAS.clip = audiosPreg[numPP]; // cargo el audio de la posición actual
            sonidoAS.Play();
            jueg[numPP].enabled = true; // activa la imagen de la posición actual.
        }
    }
}

```

Fig 122 "Fragmento de código de la función Tabulando parte 9"

```

    else if ((Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow)) || Input.GetKeyDown(KeyCode.LeftAlt))
        numPP--;//descrece el contador en 1.
    else if ((Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow)) || Input.GetKeyDown(KeyCode.LeftAlt))
        numPP++;//increce el contador en 1.

    //mismo se disminuye los audios deT, mayor cantidad detecta una entrada por teclado
    //de tipo numero menor que el anterior que estab seleccionado
    foreach (var item in audiosPregMouse)
    {
        item.Stop();
    }

    jng2[numPP].enabled = false; //desactiva la imagen de la posicion actual ->
    ap1 = audiosP1.Length - 1; //inicializa el contador en 4
    ap2 = audiosP2.Length - 1; //inicializa el contador en 4
    nextFieldPreg.numPP.Select(); //selecciona el boton en la posicion actual

    //si el numero del boton es igual a 1
    if (nextFieldPreg.numPP.name == "b_resp")
    {
        sonidoAS.Stop();
        //si el numero del boton es igual a
        if (textPreg.text == "Por favor seleccione la opción que considere correcta para seleccionar al personal que se liquidara:")
        {
            if (ap1 < 0)
            {
                sonidoAS.clip = audiosP1[ap1]; //carga el audio de la posicion actual
                sonidoAS.Play();
                ap1++; //incrementa en contador en 1
            }
        }
        else if (textPreg.text == "Ahora elija el dispositivo que usted utilizaría para comunicarse con su compañero:")
        {
            if (ap2 < 0)
            {
                sonidoAS.clip = audiosP2[ap2]; //carga el audio de la posicion actual
                sonidoAS.Play();
                ap2++; //incrementa en contador en 1
            }
        }
        //activa la imagen del boton de la posicion actual
        jng2[numPP].enabled = true;
    }
}

```

*Fig 123"Fragmento de código de la función Tabulando parte 10"*

```

750     else if (GeneralEmpezar.menu == "FinA")
751     {
752         if (botSig.activeSelf == true)//si el botón siguiente está desactivado.
753         {
754             //Si el contador numPF es menor al tamaño del arreglo nextFieldPreg.
755             if (numFF < nextFieldFin.Length - 1)//si el contador es menor al tamaño del arreglo.
756             {
757                 if (numFF == 0)//si el contador es igual a 0.
758                 {
759                     nextFieldFin[numFF].Select();//selecciono el botón en la posición actual.
760                     finI[numIFF].enabled = true;//activo la imagen de la posición actual.
761                 }
762             }
763             //Si presiona las siguientes teclas
764             if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
765             {
766                 /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
767                 y de esa manera no se cruzan los audios que estén reproduciéndose*/
768                 foreach (var item in audiosFinMouse)
769                 {
770                     item.Stop();
771                 }
772                 if (numFF == 0)// si el contador es igual a 0.
773                 {
774                     sonidoAS.Stop();
775                     sonidoAS.clip = audiosFin[numFF];//cargo el audio del arreglo audiosFin de la posición actual.
776                     sonidoAS.Play();
777                 }
778                 numFF++;//incremento el contador en 1.
779                 numIFF++;//incremento el contador en 1.
780                 nextFieldFin[numFF].Select();//selecciono el botón en la posición actual.
781                 if (numFF == 4)//si el contador es igual a 4.
782                 {
783                     if (puntajeFinal == 0)//si el puntaje final es igual a 0
784                     {
785                         sonidoAS.Stop();
786                         sonidoAS.clip = audiosFin[4];//cargo el audio del arreglo audiosFin de la posición 4
787                         sonidoAS.Play();
788                     }
789                     else if (puntajeFinal == 50)
790                     {
791                         sonidoAS.Stop();
792                         sonidoAS.clip = audiosFin[5];//carga el audio del arreglo audiosFin de la posición 5.
793                         sonidoAS.Play();
794                     }
795                     else if (puntajeFinal == 100)
796                     {
797                         sonidoAS.Stop();
798                         sonidoAS.clip = audiosFin[6];//cargo el audio del arreglo audiosFin de la posición 6
799                         sonidoAS.Play();
800                     }
801                 }
802             }
803         }
804     }

```

*Fig 124"Fragmento de código de la función Tabulando parte 11"*

```

801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819 >     else if (numFF < 4)//si el contador es menor a 4
820 {
821     sonidoAS.Stop();
822     sonidoAS.clip = audiosFin[numFF];//cargo el audio del arreglo audiosFin de la posición actual.
823     sonidoAS.Play();
824 }
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
999

```

Fig 125 "Fragmento de código de la función Tabulando parte 12"

```

925 //si menu es igual a Fin
926 else if (GeneralEmpezar.menu == "Fin")
927 {
928     if (botSig.activeSelf == true)//si el botón siguiente está activado
929     {
930         if (numFFA < nextFieldFinA.Length - 1)//si el contador numFFA es menor al tamaño del arreglo.
931         {
932             if (numFFA == 0)//si el contador es igual a 0.
933             {
934                 //selecciono el botón en la posición actual.
935                 nextFieldFinA[numFFA].Select();
936                 finAI[numIFFA].enabled = true;//activo la imagen de la posición actual.
937             }
938             //Si presiona las siguientes teclas
939             if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
940             {
941                 /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
942                 y de esa manera no se cruzan los audios que estén reproduciéndose*/
943                 foreach (var item in audiosFinAMouse)
944                 {
945                     item.Stop();
946                 }
947                 if (numFFA == 0)//si el contador es igual a 0.
948                 {
949                     sonidoAS.Stop();
950                     //cargo el audio del arreglo audiosFinA de la posición actual.
951                     sonidoAS.clip = audiosFinA[numFFA];
952                     sonidoAS.Play();
953                 }
954                 numFFA++; //incremento el contador en 1.
955                 numIFFA++; //incremento el contador en 1.
956                 //selecciono el botón en la posición actual.
957                 nextFieldFinA[numFFA].Select();
958                 sonidoAS.Stop();
959                 //cargo el audio del arreglo audiosFinA de la posición actual.
960                 sonidoAS.clip = audiosFinA[numFFA];
961                 sonidoAS.Play();
962                 //desactivo la última imagen del arreglo finAI
963                 finAI[numIFFA - 1].enabled = false;
964                 //activo la imagen de la posición actual.
965                 finAI[numIFFA].enabled = true;
966             }
967         }
968     }
969 }

```

Fig 126"Fragmento de código de la función Tabulando parte 13"

```

912     else if (Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow) || Input.GetKeyDown(KeyCode.LeftAlt))
913     {
914         //<-- Mando a detener los audios del mouse cuando detecte una tecla de flecha,
915         //y de esa manera no se crean los audios que estén reproduciéndose.
916         foreach (var item in audiosFinAMouse)
917         {
918             item.Stop();
919         }
920         if (numFFA == 0) //Si el controlador es 0
921         {
922             //Iniciamos el contador numFFA en 0 es decir seleccionamos el ultimo elemento del arreglo.
923             numFFA = nextFieldFinA.Length - 1;
924             //Iniciamos el contador numIFFA en 0 es decir seleccionamos el ultimo elemento del arreglo.
925             numIFFA = nextFieldFinA.Length - 1;
926             //Cargamos el botón en la posición actual.
927             nextFieldFinA[numFFA].Select();
928             finAI[0].enabled = false; //desactiva la imagen de la posición 0 del arreglo.
929             finAI[numIFFA].enabled = true; //activa la imagen de la posición actual.
930             sonidoAS.Stop();
931             //cargamos el audio del arreglo audiosFinA de la posición actual.
932             sonidoAS.clip = audiosFinA[numFFA];
933             sonidoAS.Play();
934         }
935         else
936         {
937             numFFA--;
938             numIFFA--;
939             //selecciona el botón en la posición actual.
940             nextFieldFinA[numFFA].Select();
941             sonidoAS.Stop();
942             //carga el audio del arreglo audiosFinA de la posición actual.
943             sonidoAS.clip = audiosFinA[numFFA];
944             sonidoAS.Play();
945             //actualiza la imagen de la posición actual.
946             finAI[numIFFA + 1].enabled = false;
947             //activa la imagen de la posición actual.
948             finAI[numIFFA].enabled = true;
949         }
950     }
951     else if (numFFA > nextFieldFinA.Length - 1) //Si el controlador es igual al tamaño del arreglo
952     {
953         //Si presiono las siguientes teclas:
954         if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
955         {
956             //<-- Mando a detener los audios del mouse cuando detecte una tecla de flecha,
957             //y de esa manera no se crean los audios que estén reproduciéndose.
958             foreach (var item in audiosFinAMouse)
959             {
960                 item.Stop();
961             }
962             finAI[numIFFA].enabled = false; //desactiva la imagen de la posición actual.
963             numFFA = 0;
964             numIFFA = 0;
965             nextFieldFinA[numFFA].Select(); //selecciona el botón en la posición actual.
966             sonidoAS.Stop();
967             sonidoAS.clip = audiosFinA[numFFA]; //cargamos el audio del arreglo audiosFinA de la posición actual.
968             sonidoAS.Play();
969             finAI[numIFFA].enabled = true; //activa la imagen de la posición actual.
970         }
971     }

```

Fig 127"Fragmento de código de la función Tabulando parte 14"

```

0023         }
0024     }
0025 
0026     //<-- cuando se detiene los audios del mouse cuando detecta una entrada por teclado,
0027     //o sea cuando se crean los audios que estan representados
0028     foreach (var item in audiosFinMouse)
0029     {
0030         item.Stop();
0031     }
0032     numFFA--;
0033     numIFFA--;
0034     nextFieldFinA[numFFA].Select(); //selecciona el boton en la posicion actual.
0035     sonidoAS.Stop();
0036     sonidoAS.Clip = audiosFinA[numFFA]; //carga el audio del arreglo audiosFinA de la posicion actual.
0037     sonidoAS.Play();
0038     finAI[numIFFA].enabled = false; //desactiva la imagen de la posicion actual <i>
0039     finAI[numIFFA].enabled = true; //activa la imagen de la posicion actual </i>.
0040 
0041     }
0042 
0043     //<-- cuando se apaga el Preferencias
0044     if (GeneralEmpezar.menu == "Preferencias")
0045     {
0046         if (prefer.activeSelf == true) //si el panel de preferencias esta activo.
0047         {
0048             //<-- el siguiente cuadro es donde se coloca del arreglo
0049             if (numCP < nextFieldPrefer.Length - 1)
0050             {
0051                 if (numCP == #) //si el contador es #
0052                 {
0053                     nextFieldPrefer[numCP].Select(); //selecciona el boton en la posicion actual.
0054                     prefImg[numFP].enabled = true; //activa la imagen de la posicion actual.
0055 
0056                     //<-- activa las siguientes teclas
0057                     if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) ||| Input.GetKeyDown(KeyCode.DownArrow))
0058                     {
0059                         //<-- cuando se detiene los audios del mouse cuando detecta una entrada por teclado,
0060                         //o sea cuando se crean los audios que estan representados
0061                         foreach (var item in audiosPreferMouse)
0062                         {
0063                             item.Stop();
0064                         }
0065                         if (numCP == #)
0066                         {
0067                             sonidoAS.Stop(); //detiene todos los audios.
0068                             sonidoAS.Clip = audiosPrefer[numCP]; //carga el audio del arreglo audiosFin de la posicion actual.
0069                             sonidoAS.Play(); //audio a reproducir el audio.
0070                             nextFieldPrefer[numCP].Select(); //selecciona el boton en la posicion actual.
0071                             prefImg[numFP + 1].enabled = false; //desactiva la ultima imagen del arreglo.
0072                             prefImg[numFP].enabled = true; //activa la imagen de la posicion actual.
0073                         }
0074                         numCP++;
0075                         numFP++;
0076                         sonidoAS.Stop(); //detiene todos los audios.
0077                         sonidoAS.Clip = audiosPrefer[numCP]; //carga el audio del arreglo audiosFin de la posicion actual.
0078                         sonidoAS.Play(); //audio a reproducir el audio.
0079                         nextFieldPrefer[numCP].Select(); //selecciona el boton en la posicion actual.
0080                         prefImg[numFP + 1].enabled = false; //desactiva la ultima imagen del arreglo.
0081                         prefImg[numFP].enabled = true; //activa la imagen de la posicion actual.
0082                     }
0083                 }
0084             }
0085         }
0086     }

```

Fig 128"Fragmento de código de la función Tabulando parte 15"

```

1135 //Si presiona las siguientes teclas
1136 else if(Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt)){
1137
1138     /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1139     y de esa manera no se cruzan los audios que estén reproduciéndose*/
1140     foreach (var item in audiosPreferMouse)
1141     {
1142         item.Stop();
1143     }
1144     if (numCP == 0)
1145     {
1146         //cambiamos el contador numCP en 11 para dar seleccionado el ultimo elemento del arreglo.
1147         numCP = nextFieldRefer.Length - 1;
1148         //cambiamos el contador numFP en 11 para dar seleccionado el ultimo elemento del arreglo.
1149         numFP = nextFieldRefer.Length - 1;
1150         nextFieldRefer[numCP].Select(); //selecciono el botón en la posición actual.
1151         sonidoAS.Stop(); //detengo todos los audios.
1152         sonidoAS.clip = audiosPrefer[numCP]; //cargo el audio del arreglo audiosFinA de la posición actual.
1153         sonidoAS.Play(); //mando a reproducir el audio.
1154         prefImg[0].enabled = false; //desactiva la imagen de la posición 0 del arreglo.
1155         prefImg[numFP].enabled = true; //activa la imagen de la posición actual.
1156
1157     }
1158
1159     else-if (numCP < nextFieldRefer.Length - 1){//si el contador es menor al tamaño del arreglo
1160
1161         numCP++; //se incrementa en 1 el contador.
1162         numFP++; //se incrementa en 1 el contador.
1163         nextFieldRefer[numCP].Select(); //selecciono el botón en la posición actual.
1164         sonidoAS.Stop(); //detengo todos los audios.
1165         sonidoAS.clip = audiosPrefer[numCP];
1166         sonidoAS.Play(); //mando a reproducir el audio.
1167         prefImg[numFP + 1].enabled = false; //desactiva la imagen de la posición actual -1.
1168         prefImg[numFP].enabled = true; //activa la imagen de la posición actual.
1169
1170     }
1171
1172     else if (numCP == nextFieldRefer.Length - 1){
1173
1174         //Si presiona las siguientes teclas
1175         if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
1176
1177             /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1178             y de esa manera no se cruzan los audios que estén reproduciéndose*/
1179             foreach (var item in audiosPreferMouse)
1180             {
1181                 item.Stop();
1182             }
1183             prefImg[numFP].enabled = false; //desactiva la imagen de la posición actual.
1184             numCP = 0;
1185             numFP = 0;
1186             sonidoAS.Stop(); //detengo todos los audios.
1187             nextFieldRefer[numCP].Select(); //selecciono el botón en la posición actual.
1188             prefImg[numFP].enabled = true; //activa la imagen de la posición actual.
1189             sonidoAS.clip = audiosPrefer[numCP]; //cargo el audio del arreglo audiosFinA de la posición actual.
1190             sonidoAS.Play();
1191
1192     }
1193
1194 }

```

Fig 129"Fragmento de código de la función Tabulando parte 16"

```

1140 //Si presiona las siguientes teclas
1141 else if(Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt)){
1142
1143     numCP--;
1144     numFP--;
1145     /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1146     y de esa manera no se cruzan los audios que estén reproduciéndose*/
1147     foreach (var item in audiosPreferMouse)
1148     {
1149         item.Stop();
1150     }
1151     prefImg[numFP+1].enabled = false;//desactiva la imagen de la posición actual.
1152     nextFieldRefer[numCP].Select(); //selecciono el botón en la posición actual.
1153     sonidoAS.Stop(); //detengo todos los audios.
1154     sonidoAS.clip = audiosPrefer[numCP]; //cargo el audio del arreglo audiosFinA de la posición actual.
1155     sonidoAS.Play();
1156     prefImg[numFP].enabled = true; //activa la imagen de la posición actual.
1157
1158 }
1159
1160 }
1161

```

Fig 130"Fragmento de código de la función Tabulando parte 17"

La función *correTiempo()* cuando al navegar ya sea mediante el mouse o teclado y este posicionado en los campos de texto iniciará la captura del tiempo de respuesta de cada actividad presentada al estudiante, dicha función es llamada en cada campo de texto de

ingreso de respuestas.

```
1162     /*Esta función me permite poner el cronómetro para poder inicializar
1163 cada vez que respondo las preguntas*/
1164     public void correTiempo()
1165     {
1166         float TimerControl = Time.time - StartTime;
1167         string mins = ((int)TimerControl / 60).ToString("00");
1168         string segs = (TimerControl % 60).ToString("00");
1169         string milisegs = ((TimerControl * 100) % 100).ToString("00");
1170         string TimerString = string.Format("{00}:{01}:{02}", mins, segs, milisegs);
1171         tiempoText.text = TimerString;//en esta variable se guarda el tiempo de respuesta empleado por pregunta.
1172     }
```

Fig 131 "Fragmento de código de la función correTiempo"

La función *Empiezo()* permite iniciar la interacción del participante con el simulador laboral, ésta función es llamada al presionar el botón empezar del panel principal, ésta a la vez llama la función nuevoJuego() para inicializar todos las variables en cero, tambien a la función GenerarAleatoriosSinRepetir() para que se generen números randomicos y mostrar las preguntas de manera aleatoria.

```
1173     //Esta función es llamada cuando se presiona el botón empezar
1174     public void Empiezo()
1175     {
1176         nuevoJuego(); //inicializa todo como un juego nuevo.
1177         textoTit.text = "EJERCITARIO INFORMACIÓN DIFÍCIL"; //se muestra en el panel main
1178         textodetall.text = "INSTRUCCIONES PARA EL PARTICIPANTE"; //se muestra en el panel main
1179         datetime = DateTime.Now.ToString("hh:mm:ss"); //registra la hora de inicio
1180         //la Imagen de la casa la cual corresponde al botón de reiniciar cambia por la imagen del botón siguiente.
1181         sigoFin.image.sprite = ImagenSiguiente;
1182         GenerarAleatoriosSinRepetir(); //llamó al metodo de generar números aleatorios.
1183         GeneralEmpezar.move = false; //deshabilita el movimiento de la cámara.
1184         empiezo.enabled = false; //desactiva el botón empezar.
1185         textoEmpiezo.gameObject.SetActive(false); //desactiva el texto del botón empezar
1186         //Activo todos los objetos del panel de instrucciones/
1187         juego.SetActive(true);
1188         botSig.SetActive(true);
1189         botPref.SetActive(true);
1190         botSalir.SetActive(true);
1191         tit.SetActive(true);
1192         detall.SetActive(true);
1193         conten.SetActive(true);
1194         GeneralEmpezar.menu = "Jugando"; //menu es igual a Jugando
1195         contadorPR = 0; //inicializo el contador en 0.
1196     }
```

Fig 132 "Fragmento de código de la función Empiezo"

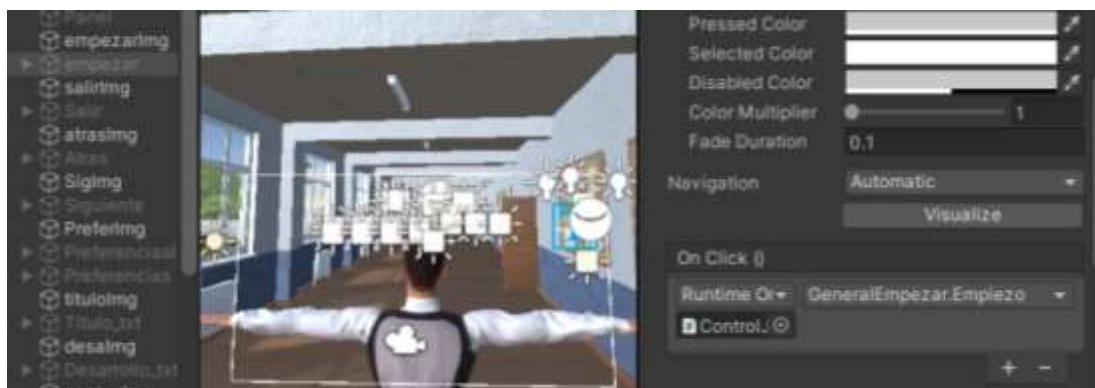


Fig 133 "Agregamos el evento On Click y llamamos a la función Empiezo"

La siguiente función es llamada dentro de la función Empezar.

```
1197     /*Función para generar números aleatorios*/
1198     void GenerarAleatoriosSinRepetir()
1199     {
1200
1201         numerosGuardados = new List<int>(); //crea una lista de numeros aleatorios guardados
1202         numPares = new List<int>(); //se crea una lista de numeros pares correspondiente a las preguntas
1203         listaFinalPregs = new List<int>(); //se crea lista de preguntas randomicas.
1204         int posicionAleatoria, numpar = 0;
1205         //si el contador es menor al tamaño de la lista contenPregFinal dividimos para dos ya que son dos el número de preguntas.
1206         for (int i = 0; i < (contenPregFinal.Count / 2); i++)
1207         {
1208             do
1209             {
1210                 //hacer que la posicionAleatoria es igual al rango de 0 a 1, en este caso/
1211                 posicionAleatoria = Random.Range(0, (contenPregFinal.Count / 2));
1212                 //mientras numerosGuardados contenga un numero de posicionAleatoria
1213                 } while (numerosGuardados.Contains(posicionAleatoria));
1214
1215                 numerosGuardados.Add(posicionAleatoria); //agrego a mi lista el número aleatorio generado.
1216                 listaFinalPregs.Add(0); //agregó a mi lista en la posición 0 el número que generé.
1217                 numPares.Add(numpar); //agrego 0 a la lista
1218                 numpar = numpar + 2; //Inicializo numpar y digo que es igual a numpar+2;
1219             }
1220             //recorro con un bucle y digo si i es menor al número de numerosGuardados +1
1221             for (int i = 0; i < numerosGuardados.Count; i++)
1222             {
1223                 //mi listaFinalPregs en la posición i es igual a el numero numero de preguntas
1224                 listaFinalPregs[i] = numPares[numerosGuardados[i]];
1225             }
1226             //recorro la lista y digo si i es menor al número de mi lista numeroGuardados +1,
1227             for (int i = 0; i < listaFinalPregs.Count; i++)
1228             {
1229                 //se agregan mis preguntas con sus alternativas.
1230                 contenPreg.Add(contenPregFinal[listaFinalPregs[i]]);
1231                 //se agregan mis preguntas con sus alternativas +1.
1232                 contenPreg.Add(contenPregFinal[listaFinalPregs[i] + 1]);
1233             }
1234         }
```

Fig 134 "Fragmento de código de la función GenerarAleatoriosSinRepetir"

La siguiente función es llamada al presionar el botón de preferencias y cuando ésta está activa se manda a desactivar todas las imágenes del resto de paneles para evitar errores.

```

1235 //Función que al llamarla mediante el botón de preferencias activa el panel de preferencias*
1236 public void ActivoPrefer()
1237 {
1238     if (pefBool == false)//si el botón de preferencias está en false
1239     {
1240         prefer.SetActive(true);//mando a activar el panel de preferencias
1241         GeneralEmpezar.menu = "Preferencias";//menu es igual a Jugando
1242         tit.SetActive(false);//desactivo el titulo para evitar que se choquen con el panel de preferencias.
1243
1244         GeneralEmpezar.pefBool = true;//se activa el botón de preferencias.
1245         if (pregu.activeSelf == true)//si mis preguntas estan activadas
1246         {
1247             /* mando a desactivar todas las imágenes del panel de preguntas*/
1248             foreach (var item in juegI)
1249             {
1250                 item.enabled = false;
1251             }
1252         }
1253         //Si el texto de la pregunta es igual a
1254         else if (textodetail.text == "INSTRUCCIONES PARA EL PARTICIPANTE")
1255         {
1256             /* mando a desactivar todas las imágenes del panel de instrucciones*/
1257             foreach (var item in dial)
1258             {
1259                 item.enabled = false;
1260             }
1261         }
1262         else if (
1263             //Si el texto de la pregunta es igual a
1264             textodetail.text == "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL"
1265         )
1266         {
1267             /* mando a desactivar todas las imágenes del panel FinA*/
1268             foreach (var item in finAI)
1269             {
1270                 item.enabled = false;
1271             }
1272         }
1273         //Si el texto de la pregunta es igual a
1274         else if (textodetail.text == "RETROALIMENTACIÓN FINAL")
1275         {
1276             /* mando a desactivar todas las imágenes del panel Fin*/
1277             foreach (var item in finI)
1278             {
1279                 item.enabled = false;
1280             }
1281         }
1282     }
1283 }

```

Fig 135"Fragmento de código de la función ActivoPrefer"



Fig 136"Agregamos el evento On Click y llamamos a la función ActivoPrefer"

La siguiente función es llamada al presionar el botón salir del panel de preferencias.

```

//función que si [salir mediante el botón salir de preferencias] desactiva el panel de preferencias
public void DesctivoPrefer()
{
    if (prefBool == true) //si el botón de preferencias está activo
    {
        prefer.SetActive(false); //desactiva el panel de preferencias
        tit.SetActive(true); //activa el título
        GeneralEmpezar.prefBool = false; //desactiva botón de
        if (pregu.activeSelf == true) //si las preguntas están activadas
        {
            GeneralEmpezar.menu = "Preguntas";
        }
        //Si el texto de la pregunta es igual a
        else if (textodetall.text == "INSTRUCCIONES PARA EL PARTICIPANTE")
        {
            GeneralEmpezar.menu = "Jugando";
        }
        else if (
            textodetall.text == "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL"
        )
        {
            GeneralEmpezar.menu = "FinA";
        }
        else if (textodetall.text == "RETROALIMENTACIÓN FINAL")
        {
            GeneralEmpezar.menu = "Fin";
        }
        //cada vez que cierra el panel de preferencias se dirigirá hacia el panel que esté activado.
    }
}

```

Fig 137"Fragmento de código de la función DesctivoPrefer"



Fig 138"Agregamos el evento On Click y llamamos a la función DesctivoPrefer"

La siguiente función es llamada dentro de la función Siguiente() y la función Atrás(), se usa para reiniciar los valores de los contados en cero cada vez que se presione le botón atrás o siguiente.

```

1314     /*Funcion que al llamarla desde la función siguiente o otras reinicia sus valores a cero*/
1315
1316     public void ReinicioSeleccion()
1317     {
1318         /*Inicializo todos los contadores en cero*/
1319         numC = 0;
1320         numI = 0;
1321         numCP = 0;
1322         numFP = 0;
1323         numPP = 0;
1324         numFPP = 0;
1325         numFF = 0;
1326         numIFF = 0;
1327         numFFA = 0;
1328         numIFFA = 0;
1329
1330         nextFieldDiag[numC].Select(); //selecciona el objeto que se encuentra en la posición
1331         nextFieldPrefe[numCP].Select(); //selecciona el objeto que se encuentra en la posición
1332         nextFieldPreg[numPP].Select(); //selecciona el objeto que se encuentra en la posición
1333         nextFieldFin[numFF].Select(); //selecciona el objeto que se encuentra en la posición
1334         nextFieldFinA[numFFA].Select(); //selecciona el objeto que se encuentra en la posición
1335
1336         /* mando a desactivar todas las imágenes del panel de preferencias/**/
1337         foreach (var item in prefImg)
1338         {
1339             item.enabled = false;
1340         }
1341         foreach (var item in dial)
1342         {
1343             item.enabled = false;
1344         }
1345         foreach (var item in juegI)
1346         {
1347             item.enabled = false;
1348         }
1349         foreach (var item in finI)
1350         {
1351             item.enabled = false;
1352         }
1353         foreach (var item in finAI)
1354         {
1355             item.enabled = false;
1356         }
1357     }

```

Fig 139"Fragmento de código de la función ReinicioSeleccion"

La siguiente función es llamada desde la función *Respondo()* y al presionar el botón siguiente, En esta función también presentan las actividades a resolver para el participante, para ello primero consultamos si *menu* es igual a *Preguntas*, si es que si entonces se muestra al participante la actividad junto a las posibles respuestas, cuando seleccione el botón la opción que crea correcta internamente se estará llamando la función *Respondo()* quien tomará las respuestas seleccionadas por el participante y las comparará contra el arreglo de respuestas correctas, una vez que el participante conteste la última pregunta pasará al panel de calificación eso lo controlamos por medio de un condicional en el que decimos si el contador de preguntas es igual al número de preguntas que tiene la lista de preguntas, significará que ha terminado de resolver las preguntas lo cual lo llevará al panel de calificación, una vez ahí se mostrará el puntaje obtenido para ello se consulta si la variable *menu* es igual a *FinA*, si es que si, entonces dice, si el puntaje obtenido es 0 la calificación es 0%, si el puntaje obtenido es 50 entonces la calificación es 50% y si el puntaje obtenido es 100 entonces tendrá el 100%; finalmente cuando la

variables *menu* sea igual a Fin entonces estaremos en el último panel, una vez que estemos internamente se llama a la función *Post()* para consumir el servicio web.

```

public void Siguiente()
{
    ReinicioSeleccion(); //Caso de la función para cuando ya no presione el botón siguiente seleccione la opción de fin.
    //Dado que el punto de preferencias está desactivado para que el siguiente punto al seleccionar la tecla siguiente.
    //este comando deberá ser el 5 punto de preferencias preferido.
    if (prefer.activeSelf == false)
    {
        TextoAgregarModif = "Nuevo";
        StartTime = Time.time; //el control de tiempo comienza en este.
        if (GeneralEmpieza.menu == "Fin") //si menu es igual a fin significa que estoy en el punto final.
        {
            GeneralEmpieza.menu = "Main"; //este menu es igual a Main significa que estoy en el punto inicio.
            //se activan las variables de inicio.
            pos = 0;
            contRes = 0;
            numC = 0;
            numL = 0;
            numCP = 0;
            numFP = 0;
            numPP = 0;
            numPPF = 0;
            numFF = 0;
            numIFF = 0;
            ap1 = 0;
            ap2 = 0;
            numProg = 0;
            puntajeFinal = 0;
            textodetail.text = "INSTRUCCIONES PARA EL PARTICIPANTE";
            //sección de instrucciones destinadas al participante
            textocanten.text = "Usted es el rector de una institución educativa privada, por la crisis ocasionada con la Pandemia Covid-19, se le visto"
        }
        TextoAgre
    }
}

```

Fig 140"Fragmento de código de la función Siguiente parte 1"





Fig 143 "Agregamos el evento On Click y llamamos a la función Siguiente"

Se crea la función *Post()* para consumir el servicio web, pasamos la url del servidor web indicando el tipo de método en este caso POST ya que se va a insertar datos en el servidor web, generamos el archivo json de mis preguntas, envío el json y finalmente hago la petición al servidor web mediante *SendWebRequest()*.



Fig 144 "Fragmento de código de la función Post()"

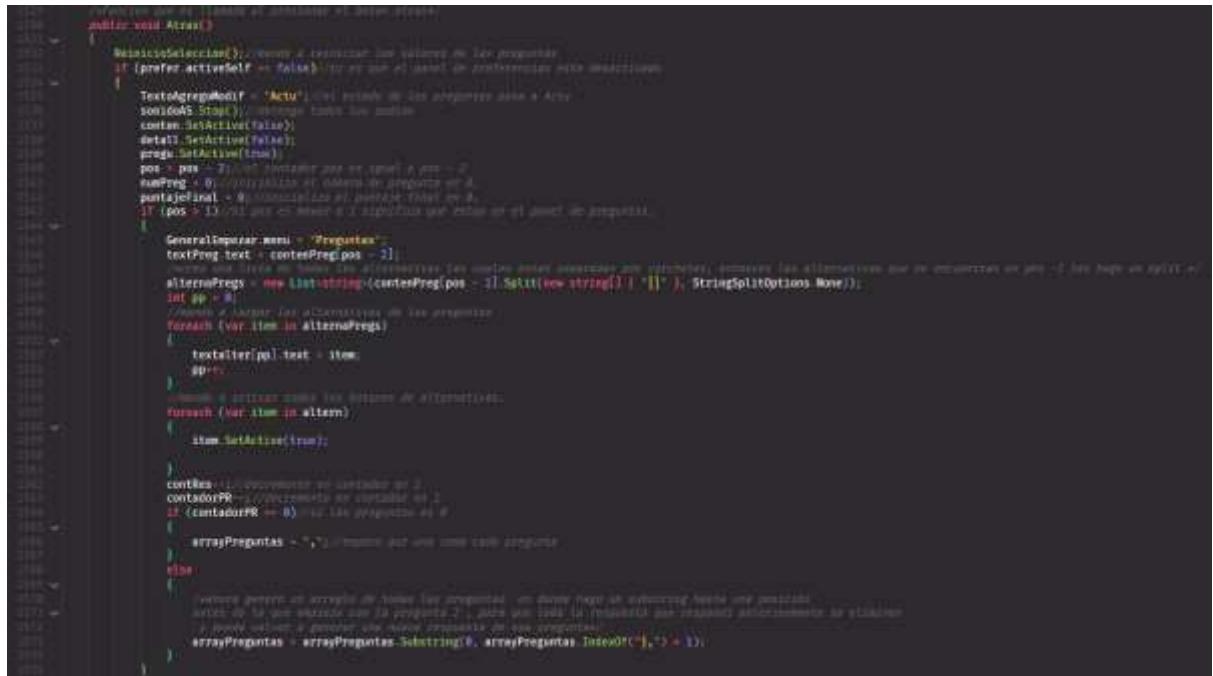


Fig 145 "Fragmento de código de la función Atrás parte 1"

```

1576
1577    else
1578    {
1579        if (pos == 0)//si el contador es 0 quiere decir que estamos en el panel de instrucciones
1580        {
1581            puntajeFinal = 0;//inicializamos el puntajeFinal en 0.
1582            //Activamos los siguientes objetos de la interfaz//
1583            juego.SetActive(true);
1584            botSig.SetActive(true);
1585            botPref.SetActive(true);
1586            botSalir.SetActive(true);
1587            botAtras.SetActive(false);//desactivo el botón atras
1588            tit.SetActive(true);
1589            detail.SetActive(true);
1590            conten.SetActive(true);
1591            arrayPreguntas = "";//limpio la variable de preguntas.
1592            GeneralEmpezar.menu = "Jugando";
1593            pregu.SetActive(false);//desactivo las preguntas
1594            //mando a desactivar todos los botones de alternativas.
1595            foreach (var item in altern)
1596            {
1597                item.SetActive(false);
1598            }
1599        }
1600    }
1601}

```

Fig 146"Fragmento de código de la función Atras parte 2"

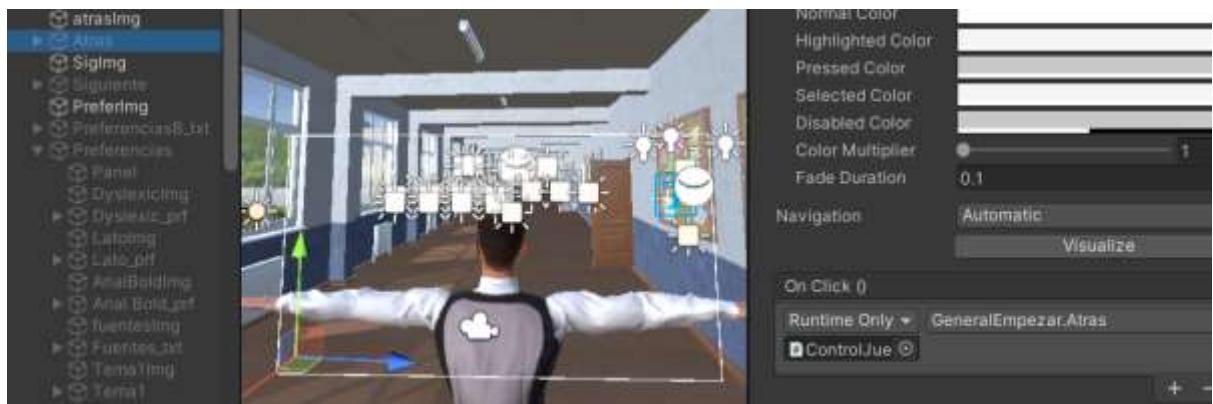


Fig 147"Agregamos el evento On Click y llamamos a la función Atras"

```

1603     public void Respondo(TextMeshProUGUI textoBot)
1604     {
1605         if (textoBot.name == rescor[contRes]) //si el nombre del texto es igual a la respuesta correcta registrada .
1606         {
1607             puntajeFinal = puntajeFinal + 50; //obtiene un puntaje de 50%.
1608         }
1609
1610         if (tiemposResp[contadorPR] == null) //si el tiempo de respuesta de mi contador es igual a null.
1611         {
1612             tiemposResp[contadorPR] = tiempoText.text; //si es igual a null le asigno un nuevo tiempo.
1613         }
1614         else
1615         { //caso contrario
1616             //creo un arreglo de string en el cual se guarda el tiempo de respuesta anterior
1617             string[] tiemResAnt = tiemposResp[contadorPR].Split(':'); //separa hora,minuto y segundo por :
1618             //creo un arreglo de string en el cual se guarda el tiempo de respuesta nuevo
1619             string[] tiemResNuevo = tiempoText.text.Split(':');
1620             //Para obtener el tiempo total de respuesta sumamos el tiemResNuevo + tiempoResAnt*/
1621             int segundos = Int32.Parse(tiemResNuevo[2]) + Int32.Parse(tiemResAnt[2]);
1622             int minutos = Int32.Parse(tiemResNuevo[1]) + Int32.Parse(tiemResAnt[1]);
1623             int horas = Int32.Parse(tiemResNuevo[0]) + Int32.Parse(tiemResAnt[0]);
1624             int auxH = 0; //auxiliar para horas
1625             int auxM = 0; //auxiliar para minutos
1626             string horF = "", minF = "", segF = "";
1627             while (segundos > 60) //mientras segundos sea menor a 60
1628             {
1629                 segundos = segundos - 60; //segundos es igual a segundos - 60
1630                 auxM++; //incremento en uno el contador.
1631             }
1632             minutos = minutos + auxM; //minutos es igual a minutos mas la auxM
1633             while (minutos > 60) //mientras minutos sea menor a 60
1634             {
1635                 minutos = minutos - 60; //minutos es igual a minutos - 60
1636                 auxH++; //incremento en uno el contador.
1637             }
1638             horas = horas + auxH; //entonces horas es igual a horas mas auxH
1639             if (horas < 10) //si horas es menor a 10
1640             {
1641                 horF = "0" + horas; //horF es igual a 0 + mas horas.
1642             }
1643             else
1644             { //caso contrario
1645                 horF = "" + horas; //se queda igual
1646             }
1647             if (minutos < 10) //si minutos es menor a 10
1648             {
1649                 minF = "0" + minutos;
1650             }
1651             else
1652             {
1653                 minF = minutos + ""; //se queda igual
1654             }
1655             if (segundos < 10) //si segundos es menor a 10
1656             {
1657                 segF = "0" + segundos;
1658             }
1659             else
1660             {
1661                 segF = segundos + ""; //se queda igual
1662             }
1663             tiemposResp[contadorPR] = horF + ":" + minF + ":" + segF; //seteo el nuevo tiempo que me demore respondiendo esa pregunta.
1664         }
}

```

Fig 148"Fragmento de código de la función Respondo parte 1"

```

1665     /*cada vez que respondo una pregunta guarda lo siguiente*/
1666     ppre = new Pregunta(); //declaro una nueva pregunta
1667     ppre.setRespuestaIngresada(textoBot.name); //seteo la respuesta ingresada
1668     ppre.setTiempoRespuesta(tiemposResp[contadorPR]); //seteo el tiempo de respuesta
1669     ppre.setNumeroPregunta(numerosGuardados[contadorPR] + 1); //seteo el numero de pregunta
1670     if (TextoAgregoModif == "Nuevo") //si estoy ingresando la respuesta textoAgregoModif es Nuevo.
1671     {
1672         eje.addPregunta(ppre); //agrego la pregunta al archivo json
1673     }
1674     else if (TextoAgregoModif == "Actu") //si presioné el boton atras textoAgregoModif es Actu
1675     {
1676         eje.updatePregunta(ppre, eje.preguntas[contadorPR]); //actualizo la respuesta sólo en esa posición
1677     }
1678
1679     sonidoAS.Stop(); //como voy a pasar al siguiente panel detengo todos los audios.
1680     contRes++; //el contador de las respuestas aumenta en 1.
1681     numPreg++; //el numero de la pregunta aumenta en 1.
1682     numPP = 0; //inicializo en 0 el contador de preguntas.
1683     numFPP = 0; //inicializo en 0 el contador de imágenes de las preguntas.
1684     contadorPR++; //incremento en 1 el contador.
1685     Siguiente(); //llamo a la función siguiente para que me de paso a la siguiente pregunta.
1686 }

```

Fig 149"Fragmento de código de la función Respondo parte 2"

La función *MenuSel()* permite controlar en que pantalla estamos en ese momento, en donde la variable *menu* va cambiando según el panel en el que esté, es decir cuando menu es igual a Main significa que está en el panel principal o main, cuando menu sea igual Jugando es que estamos en el panel de instrucciones, cuando menu es igual a Preguntas es que esta en el panel de Preguntas, cuando menu es igual a FinA es que esta en el panel de calificación y cuando menu es igual a Fin es que está en último panel en donde se muestra la retroalimentación del ejercitario resuelto.

```
744 } //método para controlar en que menu o panel me encuentro.  
745 public static void MenuSel(string tipoMen)  
746 {  
747     GeneralEmpezar.menu = tipoMen;  
748 }  
749 } //fin de clase GeneralEmpeza |
```

Fig 150"Fragmento de código de la función MenuSel"

#### 1.2.1.3 Declaración de clases serializadas

- **Clase Serializada Pregunta**

La clase serializada Pregunta hace referencia a la estructura de cada pregunta, es decir cada vez que se responda una pregunta, se guardarán los campos de datos correspondiente a la pregunta.

```
[Serializable]
/*Clase que hace referencia a la estructura de cada pregunta*/
public class Pregunta
{
    public string respuestaIngresada;
    public string tiempoRespuesta;
    public int numeroPregunta;
    public DateTime inicio;
    public DateTime fin;
    public string getRespuestaIngresada()
    {
        return respuestaIngresada;
    }
    public string getTiempoRespuesta()
    {
        return tiempoRespuesta;
    }
    public void setNumeroPregunta(int numeroPregunta)
    {
        this.numeroPregunta = numeroPregunta;
    }
    public int getPregunta()
    {
        return numeroPregunta;
    }
    public void setInicio()
    {
        this.inicio = System.DateTime.Now;
    }
    public void setFin()
    {
        this.fin = System.DateTime.Now;
    }
    public void setTiempoRespuesta(string tiempoRespuesta)
    {
        this.tiempoRespuesta = tiempoRespuesta;
    }
    public void setRespuestaIngresada(string respt)
    {
        this.respuestaIngresada = respt;
    }
}
```

Fig 151 "Fragmento de código de la clase serializada Pregunta"

La clase serializada SimuladorData hace referencia al json que se va a generar, ya que un simulador laboral puede tener n preguntas, se crea una lista de preguntas.

```
public class SimuladorData
{
    /*atributos de archivo json*/
    public string correo;
    public int numeroEjercitario;
    public string tiempoInicio;
    public string tiempoFin;
    public string fechaDeActividad;
    /* Ya que un simulador puede tener n preguntas se crea una lista de preguntas. */
    public List<Pregunta> preguntas = new List<Pregunta>();

    public void ReinicioPreguntas()
    {
        /* Se crea una lista del objeto preguntas. */
        preguntas = new List<Pregunta>();
    }

    public void setNumeroEjercitario(int numero)
    {
        this.numeroEjercitario = numero;
    }

    public int getNumeroEjercitario()
    {
        return this.numeroEjercitario;
    }

    public void setTiempoInicio(string tiempoInicio)
    {
        this.tiempoInicio = tiempoInicio;
    }

    public string getTiempoInicio()
    {
        return this.tiempoInicio;
    }

    public void setTiempoFin(string tiempoFin)
    {
        this.tiempoFin = tiempoFin;
    }

    public string getTiempoFin()
    {
        return this.tiempoFin;
    }

    public void setFechaActividad(string fechaDeActividad)
    {
        this.fechaDeActividad = fechaDeActividad;
    }

    public string getFechaActividad()
    {
        return this.fechaDeActividad;
    }

    public void setCorreo(string correo)
    {
        this.correo = correo;
    }

    public string getCorreo()
    {
        return this.correo;
    }

    public void addPregunta(Pregunta nuevaPreg)
    {
        this.preguntas.Add(nuevaPreg);
    }

    public void updatePregunta(Pregunta upPreg, Pregunta antPreg)
    {
        this.preguntas.Remove(antPreg);
        this.preguntas.Add(upPreg);
    }

    public Pregunta readPregunta(int indice)
    {
        return preguntas[indice];
    }

    public void setInicioPregunta(int numero)
    {
        this.preguntas[numero].setInicio();
    }
}
```

Fig 152 "Fragmento de código de la clase serializada SimuladorData"

```

{
  "correo": "naraujop@gmail.com",
  "numeroEjercitario": 10,
  "tiempoInicio": "03:17:03",
  "tiempoFin": "03:17:12",
  "fechaDeActividad": "2022-10-22 03:17:12"
  , "preguntas": [{"respuestaIngresada": "1",
    "tiempoRespuesta": "00:03:86",
    "numeroPregunta": 1},
    {"respuestaIngresada": "1",
    "tiempoRespuesta": "00:02:68",
    "numeroPregunta": 2}]}

```

*Fig 153"Estructura de preguntas que se genera en el archivo json"*

#### 1.4.2 Script MouseOver

##### 1.4.2.1 Declaración de variables y relación con objetos de interfaz de Unity

El siguiente script permite que cuando el mouse está sobre un botón se detengan todos los sonidos que hayan estado reproduciéndose y solo me reproduce el audio que esté en dicho botón y detiene todo cuando el mouse sale del botón.

```

1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  using UnityEngine.Events;
5
6  public class MouseOver : MonoBehaviour, IPointerEnterHandler, IPointerExitHandler
7  /*Además de la extensión MonoBehaviour se debe agregar la extensión
8  IPointerEnterHandler(mientras el mouse esté sobre el objeto) y
9  IPointerExitHandler(cuando el mouse salga del objeto)*/
10 {
11
12     /*Se declara variable booleana para controlar la reproducción
13     de audio al pasar el mouse sobre el botón*/
14     public bool isOver = false;
15
16     /*audioplayer audio del objeto que va a sonar y audioplayer del
17     objeto generalTextos para que todos los audios se detengán*/
18     public AudioSource audio, generalTextos;
19     //El sonido que se carga para que suene
20     public AudioClip soni;
21
22     /*Se hace uso del método por defecto el cual permite reproducir el
23     audio de un botón cuando este esté sobre él*/
24     public void OnPointerEnter(PointerEventData eventData)
25     {
26         //detiene todos los audios
27         audio.Stop();
28         //detiene los sonidos generales
29         generalTextos.Stop();
30         //lo a decir que está sobre el objeto
31         isOver = true;
32         audio.clip = soni;//agrega el audio
33
34         audio.Play(); //reproduce el audio
35     }
36
37     /*Se hace uso del método por defecto el cual permite dejar de reproducir cuando el mouse ya no esté sobre el botón*/
38     public void OnPointerExit(PointerEventData eventData)
39     {
40         isOver = false;
41         audio.Stop(); //detiene todos los audios.
42         generalTextos.Stop(); //detiene los sonidos generales
43     }
44 }

```

*Fig 154"Código fuente de script MouseOver"*



Fig 155 "Relación de objeto con la variable declarada"

#### 1.4.3 Script OverTextos.cs

Este script permite detener todos los audios que estén reproduciéndose y reproduce el audio de un botón cuando el mouse está sobre él, y cuando el mouse sale del botón se detiene todo.

##### 1.4.4.1 Declaración de variables

```
Assets > Scripts > OverTextos.cs
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  using UnityEngine.Events;
5  using UnityEngine.UI;
6  using TMPro;
7
8  public class OverTextos : MonoBehaviour, IPointerEnterHandler, IPointerExitHandler
9  {
10
11     public bool IsOver = false;
12     public AudioClip[] audios;
13     public AudioSource source, generalTextos; //audiosource audio del objeto que va a sonar y AudioSource del objeto generalTextos para que todos los audios se detengan
14     [SerializeField]
15     private TextMeshProUGUI texto; //declaro una variable de tipo texto
```

Fig 156 "Declaración de variables"

##### 1.4.4.2 Declaración de funciones

Esta función ocurre cuando el mouse está sobre el botón.

```

//Cuando el mouse este sobre el botón //
public void OnPointerEnter(PointerEventArgs eventData)
{
    source.Stop(); //detiene todos los audios
    generalTextos.Stop(); //detiene todos los audios
    if(GeneralEmpezar.menu == "Jugando"){
        if(texto.text == "EJERCITARIO INFORMACIÓN DIFÍCIL"){// y si el texto que estoy seleccionando es igual a este
            source.clip = audis[0]; //cargo el audio que esté en la posición 0.
            source.Play(); //mando a reproducir el audio.
        }
        else if(texto.text == "INSTRUCCIONES PARA EL PARTICIPANTE"){
            source.clip = audis[1]; //cargo el audio que esté en la posición 1.
            source.Play();
        }
        // si el texto que estoy seleccionando es igual a este
        else if(texto.text == "Usted es el rector de una institución educativa privada, por la crisis ocasionada con la Pandemia Covid 19, se ha visto muy afectado por la situación, por lo tanto, le pedimos que nos ayude a mejorar la situación." ){
            source.clip = audis[2]; //cargo el audio que esté en la posición 2.
            source.Play();
        }
    }
    else if(GeneralEmpezar.menu == "FinA"){//si menu es igual a FinA
        if(texto.text == "EJERCITARIO INFORMACIÓN DIFÍCIL"){// y si el texto que estoy seleccionando es igual a este
            source.clip = audis[0]; //cargo el audio que esté en la posición 0.
            source.Play();
        }
        // y si el texto que estoy seleccionando es igual a este
        else if(texto.text == "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL"){
            source.clip = audis[3]; //cargo el audio que esté en la posición 3.
            source.Play();
        }
        else if(texto.text.Contains(" 0% ")){//ya que es muy largo el texto que necesito comparar, lo que se hace es mandar a buscar si contiene 0%
            source.clip = audis[4]; //cargo el audio que esté en la posición 4.
            source.Play();
        }
        else if(texto.text.Contains(" 50% ")){//mando a buscar si contiene 50%
            source.clip = audis[5]; //cargo el audio que esté en la posición 5.
            source.Play();
        }
        else if(texto.text.Contains(" 100% ")){//mando a buscar si contiene 100%
            source.clip = audis[6]; //cargo el audio que esté en la posición 6.
            source.Play();
        }
    }
    else if(GeneralEmpezar.menu == "Fin"){//si menu es igual a Fin
        if(texto.text == "EJERCITARIO INFORMACIÓN DIFÍCIL"){// y si el texto que estoy seleccionando es igual a este
            source.clip = audis[0]; //cargo el audio que esté en la posición 0.
            source.Play();
        }
        else if(texto.text == "RETROALIMENTACIÓN FINAL"){
            source.clip = audis[10]; //cargo el audio que esté en la posición 10.
            source.Play();
        }
        else if(texto.text == "Competencia toma de decisiones\n\nPara un proceso de desvinculación de personal es necesario seleccionar la opción más objetiva y oportuna para cada situación."){
            source.clip = audis[9]; //cargo el audio que esté en la posición 9.
            source.Play();
        }
    }
    if(GeneralEmpezar.menu == "Preguntas"){
        if(texto.text == "EJERCITARIO INFORMACIÓN DIFÍCIL"){// y si el texto que estoy seleccionando es igual a este
            source.clip = audis[0]; //cargo el audio que esté en la posición 0.
            source.Play();
        }
        // si el texto que estoy seleccionando es igual a este
        if(texto.text == "Por favor seleccione la opción que considere correcta para seleccionar al personal que se liquidará." ){
            source.clip = audis[7]; //cargo el audio que esté en la posición 7.
            source.Play();
        }
        // si el texto que estoy seleccionando es igual a este
        else if(texto.text == "Ahora elija el discurso que usted utilizaría para comunicar a su compañero:"){
            source.clip = audis[8]; //cargo el audio que esté en la posición 8.
            source.Play();
        }
    }
}

```

Fig 157"Funcion OnPointerEnter cuando el mouse está sobre el botón"

Esta función ocurre cuando el mouse sale del botón.

```

4
5     public void OnPointerExit(PointerEventArgs eventData)
6     {
7         //quito el mouse y se detiene todo
8         isOver = false;
9         source.Stop(); //detendrá todos los audios.
10        generalTextos.Stop();
11    }
12

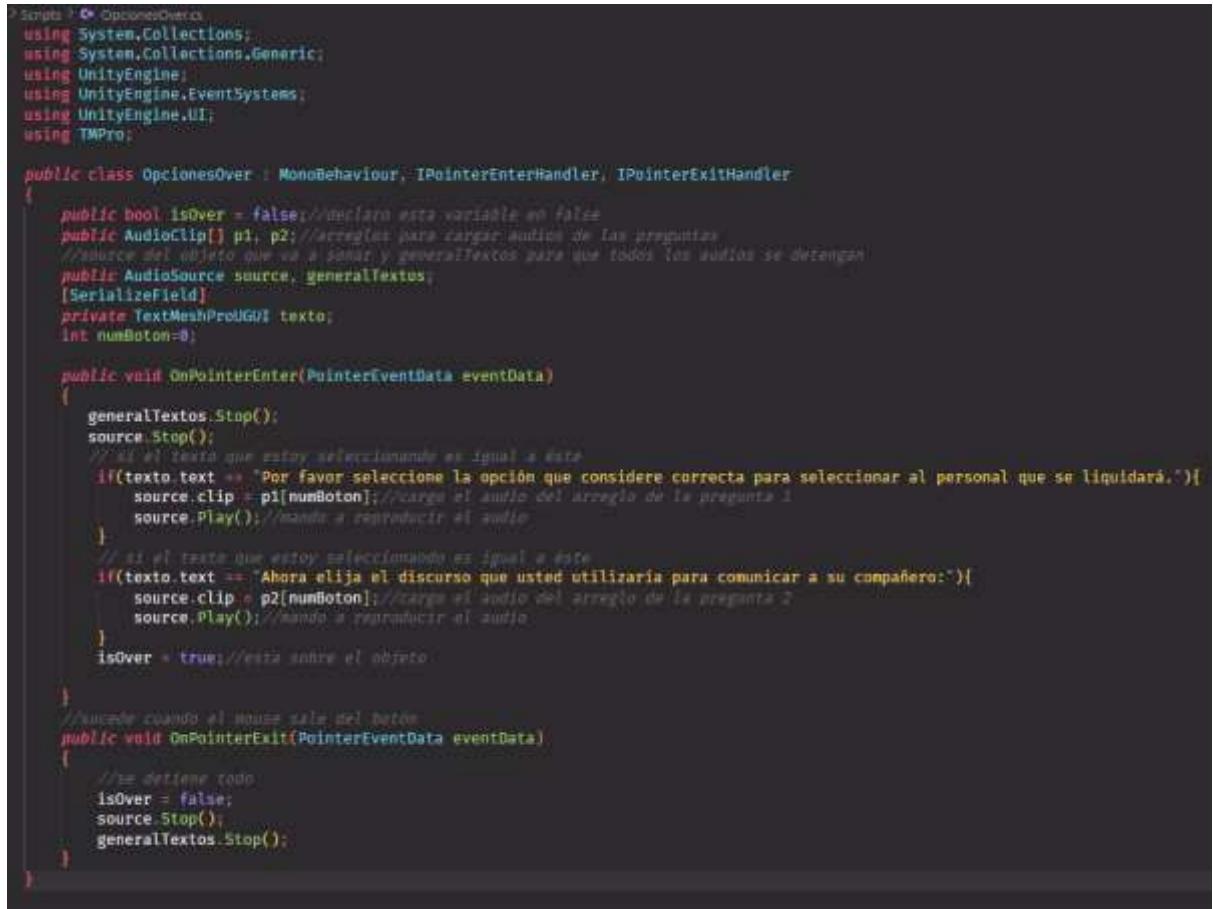
```

Fig 158"Función OnPointerExit cuando el mouse sale del botón"

#### 1.4.4 Script OpcionesOver.cs

El siguiente script permitirá que cuando el mouse esté sobre un botón reproduzca un audio, en este caso se utiliza los botones correspondientes a las alternativas de cada pregunta, para ello dentro de la función `OnPointerEnter()` usamos un condicional en donde le decimos que reproduzca el audio de dicho botón solo si el texto del enunciado de la pregunta es igual al texto que estamos mandando a comparar.

##### 1.4.5.1 Declaración de variables y funciones



```
/* Scripts / OpcionesOver.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Events;
using UnityEngine.UI;
using TMPro;

public class OpcionesOver : MonoBehaviour, IPointerEnterHandler, IPointerExitHandler
{
    public bool isOver = false; // declaro esta variable en false
    public AudioClip[] p1, p2; // arrays para cargar audio de las preguntas
    // source del objeto que va a sonar y generalTextos para que todos los audios se detengan
    public AudioSource source, generalTextos;
    [SerializeField]
    private TextMeshProUGUI texto;
    int numBoton=0;

    public void OnPointerEnter(PointerEventData eventData)
    {
        generalTextos.Stop();
        source.Stop();
        // si el texto que estoy seleccionando es igual a este
        if(texto.text == "Por favor seleccione la opción que considere correcta para seleccionar al personal que se liquidará:")
            source.clip = p1[numBoton]; //carga el audio del arreglo de la pregunta 1
            source.Play(); //manda a reproducir el audio
        else
            // si el texto que estoy seleccionando es igual a este
            if(texto.text == "Ahora elija el discurso que usted utilizaría para comunicar a su compañero:")
                source.clip = p2[numBoton]; //carga el audio del arreglo de la pregunta 2
                source.Play(); //manda a reproducir el audio
        }
        isOver = true; //esta sobre el objeto
    }

    // sucede cuando el mouse sale del botón
    public void OnPointerExit(PointerEventData eventData)
    {
        //se detiene todo
        isOver = false;
        source.Stop();
        generalTextos.Stop();
    }
}
```

Fig 159"Script OpcionesOver"

#### 1.4.5 Script MouseColor.cs

Este script permite resaltar los objetos cuando el mouse este sobre alguno de ellos, es decir, cuando el mouse esté sobre un botón éste cambiara de color, de esta manera el estudiante sabrá exactamente en donde está posicionado, para ello se ha implementado dos funciones, cambiar de color cuando el mouse esté sobre algún botón (función `OnPointerEnter ()`) y cuando el mouse salga de dicho botón se activará la función `OnPointerExit ()` volviendo al color anterior del botón.

#### 1.4.6.1 Declaración de variables y relación con objetos de interfaz de Unity

```
public class MouseColor : MonoBehaviour, IPointerEnterHandler, IPointerExitHandler
{
    //declaro esta variable en false, para que solo se active cuando pase el mouse sobre el objeto
    public bool isOver = false;
    public Image botonAct; //declaro una variable de tipo Image
```

Fig 160 "Declaración de variables"



Fig 161 "Asignamos un botón a la variable BotonAct"

```
13     //esta función sucede cuando el mouse esta sobre el objeto
14     public void OnPointerEnter(PointerEventData eventData)
15     {
16         isOver = true;
17         //si colorBt es Blanco
18         if(Preferscript.colorBt == "Blanco"){
19             //accedo a la propiedad color del botón
20             var temColor = botonAct.color;
21             temColor.a = 0.5f;
22             temColor.r = 0.52f;
23             temColor.g = 0.5f;
24             temColor.b = 0.5f;
25             botonAct.color = temColor; //le asigna un nuevo color
26         }
27         //si el nombre del botón es Tema1
28         else if (botonAct.name == "Tema1"){
29             var temColor = botonAct.color;
30             temColor.a = 0.5f;
31             temColor.r = 0.52f;
32             temColor.g = 0.5f;
33             temColor.b = 0.5f;
34             botonAct.color = temColor; //le asigna un nuevo color
35         }
36         else{
37             //accedo a la propiedad color del botón
38             var temColor = botonAct.color;
39             temColor.a = 0.75f;
40             temColor.r = 0.52f;
41             temColor.g = 0.5f;
42             temColor.b = 0.5f;
43             botonAct.color = temColor;
44         }
45     }
```

Fig 162 "Función sucede cuando el mouse está sobre el objeto"

```

46 //esta función sucede cuando el mouse sale del objeto
47 public void OnPointerExit(PointerEventData eventData)
48 {
49     isOver = false;
50     //si el nombre del botón es 12, 24, Dyslexic,lató o ArialBold
51     if(botonAct.name == "12" || botonAct.name == "24" || botonAct.name == "Dyslexic" || botonAct.name == "Lató" || botonAct.name == "Arial Bold"){
52         if(Preferscript.colorBt == "Blanco"){//si colorBt es Blanco
53             botonAct.color = Color.white;//le asigna un nuevo color
54         }
55     }else{//le asigna un nuevo color
56         botonAct.color = Color.black;
57     }
58     //si el nombre del botón contiene _acc
59     if(botonAct.name.Contains("_acc")){
60         //le asigna el color a todos los objetos que contengan ese nombre
61         botonAct.color=Color.white;
62     }
63     //si el nombre del botón es Tema1
64     else if (botonAct.name == "Tema1"){
65         //le asigna un nuevo color
66         botonAct.color = Color.white;
67     }
68     //si el nombre del botón es Tema2
69     else if (botonAct.name == "Tema2"){
70         //le asigna un nuevo color
71         botonAct.color = Color.black;
72     }
73     //si el nombre del botón contiene _prf
74     else if(botonAct.name.Contains("_prf")){
75         //si colorBt es Blanco
76         if(Preferscript.colorBt == "Blanco"){
77             //le asigna un nuevo color
78             botonAct.color=Color.white;
79         }
80         //si colorBt es Negro
81         else if(Preferscript.colorBt == "Negro"){
82             //le asigna un nuevo color
83             botonAct.color=Color.black;
84         }
85     }
86     //si el nombre del botón contiene _txt
87     else if(botonAct.name.Contains("_txt")){
88         var temColor = botonAct.color;
89         temColor.a = 0;
90         temColor.r = 255;
91         temColor.g = 255;
92         temColor.b = 255;
93         //le asigna un nuevo color
94         botonAct.color = temColor;
95     }
96     //si el nombre del botón contiene _resp
97     else if(botonAct.name.Contains("_resp")){
98         //si colorBt es Blanco
99         if(Preferscript.colorBt == "Blanco"){
100             //le asigna un nuevo color
101             botonAct.color = Color.white;
102         }
103         else{
104             //le asigna un nuevo color
105             botonAct.color = Color.black;
106         }
107     }
108     //si colorBt es Blanco
109     else if(Preferscript.colorBt == "Blanco"){
110         //le asigna un nuevo color
111         botonAct.color = Color.white;
112     }
113     else{
114         //le asigna un nuevo color
115         botonAct.color = Color.yellow;
116     }
117 }
118 }
```

Fig 163 "Esta función sucede cuando el mouse sale del objeto"

#### 1.4.6 Script PreferScrip.cs

El siguiente script contiene opciones de preferencia con las cuales el participante puede personalizar el simulador laboral de acuerdo a sus necesidades y capacidades; para las opciones de contraste se emplea tres funciones las cuales trabajan en conjunto (*Cambio()*, *CambioColBot()*, *CambioColText()*) con las cuales podrá cambiar el color de fondo de la pantalla, el color de los botones y el color del texto; para la opción de activar o desactivar la reproducción de audios se implementa la función *MuteAll()*, para la opción de tipos de fuentes se implementa la función *cambioFuente()* para ello se tiene tres

opciones de fuentes; para la opción de aumentar o reducir el tamaño de texto se implementa la función *cambioTamaño()*.

#### 1.4.3.1 Declaración de variables y relación con objetos de interfaz de Unity

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;
using UnityEngine.UI;
using System.IO;

public class PreferScript : MonoBehaviour
{
    [SerializeField]
    private Image[] botones; // arreglo de imágenes utilizadas para combinar el valor de los botones
    [SerializeField]
    // arreglo de textos utilizados para cambiar el color del texto,
    // tanto como de color y tamaño de fuente.
    private TextMeshProUGUI[] objeto, prefs, objPrefer, objText;
    [SerializeField]
    // arreglo de tipos el cual contiene las fuentes a utilizar
    private TextMeshProUGUI[] tipoLetra;
    [SerializeField]
    // arreglo de tipos el cual contiene los títulos de contrastes a usar
    private TextMeshProUGUI tem1, tem2;
    [SerializeField]
    // arreglo de imágenes el cual contiene el panel,
    // el botón de activación mute y el panel de preferencias
    private Image panel, buttonMute;
    // variable de tipo botón
    public Button butMute;
    public AudioSource[] sonidoAS;
    // variable de tipo texto del botón silenciar
    public TextMeshProUGUI silbta;
    // variable utilizada para controlar en que modo estamos
    public static string menu = "Main";
    // variable utilizada para controlar el tipo de contraste
    public static string colorBT = "Negro";
    // variable utilizada para controlar el audio
    public static bool isMute = false;
    [SerializeField]
    // crea un arreglo de imágenes de los botones principales
    private Image[] botonesImagenes;
    // crea otro arreglo para controlar el activar y desactivar el audio
    public Sprite ImageMute, ImageMutes;
    // variable booleana para controlar cuento la fuente Dyslexic esta activa
    private bool Dislex=false;
}
```

Fig 164"Fragmento de código de declaración de variables"

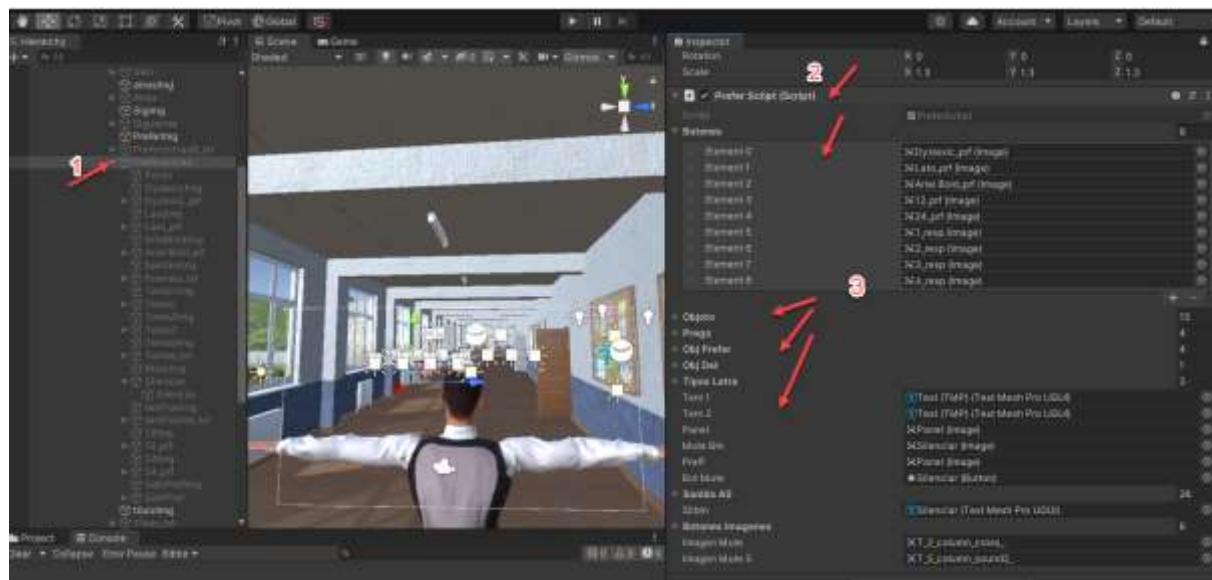


Fig 165"Asociamos los componentes con las variables declaradas"

```

45  /*Esta función recibe como parámetro un string en el cual indicamos el color del panel*/
46  public void Cambio(string colpan){
47      if (colpan == "Blanco"){//si el string que recibe es igual a Blanco
48          panel.color = new Color(255,255,255,100);//cambiamos el color del panel a blanco
49          preff.color=new Color(50,50,50,150); //cambiamos el color del panel de preferencias a oscuro.
50      } // si el string que recibe es igual a Negro
51      else if (colpan == "Negro"){
52          panel.color = new Color(0,0,0,215); //cambiamos el color del panel a negro
53          preff.color=Color.grey; //cambiamos el color del panel de preferencias a gris.
54      }
55  }
56

```

Fig 166"Función para cambiar el color del panel"

Las funciones *Cambio()*, *CambioColText()* y *CambioColBot()* son llamadas al presionar el botón tema1, para ello agregamos la propiedad *OnClick()* a nuestro botón y agregamos las funciones necesarias a las cuales les pasamos los parámetros que se necesita.



Fig 167"Combinación de funciones en donde se tiene como resultado un contraste claro"

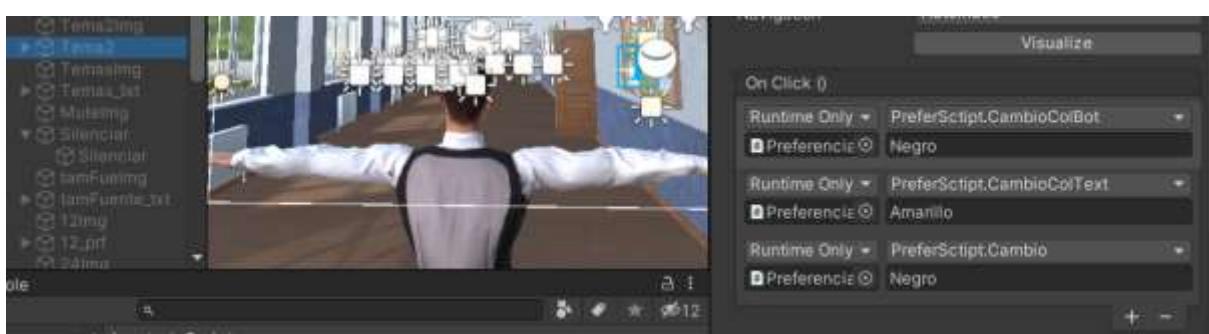


Fig 168"Combinación de funciones en donde se tiene como resultado un contraste oscuro"

```

57  /*Función que permite silenciar la salida de audio del simulador*/
58  public void MuteAll(){
59      //si la variable estatica es igual a false y colorBt es blanco
60      if(isMute == false && PreferScrip.colorBt == "Blanco"){
61          //mandamos a silenciar todos los audios.
62          foreach (var s in sonidoAS){
63              s.volume = 0f;
64          }
65          isMute = true;
66          //si esta desactivado el audio asignamos la imagen ImagenMute al botón
67          botMute.image.sprite = ImagenMute;
68          muteBtn.color = Color.yellow;//cambiamos a amarillo
69      }
70      //si la variable estatica es igual a false y colorBt es Negro
71      else if(isMute == false && PreferScrip.colorBt == "Negro"){
72          //mandamos a silenciar todos los audios.
73          foreach (var s in sonidoAS){
74              s.volume = 0f;
75          }
76          isMute = true;
77          botMute.image.sprite = ImagenMute;
78          muteBtn.color = Color.white;
79      }
80      else{//caso contrario
81          //mandamos a activar la salida de audios.
82          foreach (var s in sonidoAS){
83              s.volume = 1f;
84          }
85          isMute = false;
86          //Si colorBt es blanco
87          if(PreferScrip.colorBt == "Blanco"){
88              //cambiamos la imagen
89              botMute.image.sprite = ImagenMuteS;
90              //cambiamos a blanco
91              muteBtn.color = Color.white;
92          }
93          else{
94              botMute.image.sprite = ImagenMuteS;
95              //caso contrario lo cambia a amarillo
96              muteBtn.color = Color.yellow;
97          }
98      }
99  }
100 }
```

Fig 169"Función para silenciar la salida de audios"



Fig 170"Función que permite silenciar los audios."

Función llamada desde el botón tema1 y tema2.

```

101  /*Función que recibe como parametro un string dicha función permite cambiar el color de los botones*/
102  public void CambioColBot(string te){
103      //si el string que recibe es igual a Blanco
104      if (te == "Blanco"){
105          //cambia todos los botones a blanco
106          foreach(var b in botones){
107              b.color = Color.white;
108              PreferSctipt.ColB("Blanco");
109          }
110          foreach(var b in botonesImagenes){
111              b.color = Color.white;
112          }
113          foreach(var b in pregis){
114              b.color = Color.white;
115          }
116      }
117      //si el string que recibe es igual a Negro
118      else if (te == "Negro"){
119          //cambia todos los botones a negro
120          foreach(var b in botones){
121              b.color = Color.black;
122              PreferSctipt.ColB("Negro");
123          }
124          //los botones principales los cambiamos a amarillo
125          foreach(var b in botonesImagenes){
126              b.color = Color.yellow;
127          }
128      }
129  }
130

```

Fig 171 "Función para cambiar el color de los botones"

La siguiente función es llamada al presionar el botón tema1 y tema2.

```

131  /*Esta función recibe como parametro un string dicha función permite cambiar el color del texto*/
132  public void CambioColText(string te){
133      //si el string que recibe es igual a Negro
134      if (te == "Negro"){
135          /*cambia la letra a azul*/
136          foreach (var obj in objeto) {
137              obj.color= Color.blue;
138          }
139          foreach (var obj in tiposLetra) {
140              obj.color= Color.blue;
141          }
142          foreach (var obj in pregis) {
143              obj.color= Color.blue;
144          }
145          foreach (var obj in objPrefer) {
146              obj.color= Color.blue;
147          }
148      }
149      //si el string que recibe es igual a Amarillo
150      if (te == "Amarillo"){
151          /*cambia la letra a amarillo*/
152          foreach (var obj in objeto) {
153              obj.color = Color.yellow;
154          }
155          foreach (var obj in tiposLetra) {
156              obj.color = Color.yellow;
157          }
158          foreach (var obj in pregis) {
159              obj.color= Color.yellow;
160          }
161          foreach (var obj in objPrefer) {
162              obj.color=Color.yellow;
163          }
164      }
165  }
166

```

Fig 172 "Función para cambiar el color del texto"

La siguiente función es llamada al presionar los botones para cambiar el tipo de fuente(Dyslexic, Lato y Arial Bold).

```
268 //Esta función permite indicar como parametro un string el cual se le pasa la fuente, esta función  
269 //permite personalizar el texto de la interfaz haciendo uso de tres tipos de fuente de texto.  
270 public void cambioFuente(TMP_FontAsset fu){  
271     //Si el fuente que recibes igual a:  
272     if(fu.ToString() == "OpenDyslexic-Regular SDF (TMPPro.TMP_FontAsset)"){  
273         //se define un tamaño para cuando este activa esta fuente.  
274         foreach(var t in objeto){  
275             t.fontSize = 19f;  
276         }  
277         foreach(var t in pregis){  
278             t.fontSize = 18f;  
279         }  
280         foreach(var t in objPrefer){  
281             t.fontSize = 17f;  
282         }  
283     }  
284     Disle=true;//si está activa  
285 }  
286 else{  
287     Disle=false;// si está desactivada  
288 }  
289 //cambia de fuente a todos los textos de la interfaz.  
290 foreach(var t in objeto){  
291     t.font = fu;  
292 }  
293 foreach(var t in pregis){  
294     t.font = fu;  
295 }  
296 foreach(var t in objPrefer){  
297     t.font = fu;  
298 }  
299 //cambia el tipo de fuente  
300 tem1.font = fu;  
301 tem2.font = fu;
```

Fig 173"Función para cambiar el tipo de fuente"



Fig 174"Función que permite cambiar el tipo de fuente"

La siguiente función es llamada con los botones de aumentar y disminuir el tamaño de texto.

```

244     /*Función que recibe como parametro un entero dicha función permite reducir y aumentar el tamaño de texto*/
245     public void cambioTamano(int tam){
246
247         if(Diseño==true){ //si este activo esta la fuente
248             //nos cambia el tamaño de los estilos -2 a 4
249             foreach(var t in objeto){
250                 t.fontSize = tam-2;
251             }
252             //nos cambia el tamaño de los estilos -2 a 4 objetos de preferencias
253             foreach(var t in objPrefer){
254                 t.fontSize = tam-2;
255             }
256             //nos cambia el tamaño de los estilos -2 a 4 preguntas
257             foreach(var t in preg){
258                 t.fontSize+= (tam-2);
259             }
260             //cambia el tamaño
261             tem1.fontSize = tam;
262             tem2.fontSize = tam;
263         }
264
265         else{ //si Diseño es igual a False
266             //cambiamos el tamaño de texto a todos los objetos
267             foreach(var t in objeto){
268                 t.fontSize = tam;
269             }
270             foreach(var t in preg){
271                 t.fontSize = tam;
272             }
273             foreach(var t in objPrefer){
274                 t.fontSize = tam;
275             }
276             //cambia el tamaño
277             tem1.fontSize = tam;
278             tem2.fontSize = tam;
279         }
280     }

```

Fig 175"Función para cambiar aumentar y disminuir el tamaño de texto"



Fig 176"Funciona que permite cambiar el tamaño de texto"

Esta función es llamada al presionar el botón de preferencias.

```

242     /*Función que permite detener todos los sonidos cuando se desactive el audio y
243     así evitar que algún audio se reproduzca hasta finalizar*/
244     public void DetengoSonidos(){
245         foreach(var t in sonidoAS){
246             t.Stop();
247         }
248     }

```

Fig 177"Función para detener la salida de audio inmediatamente"

```

249     /*Función que recibe como parametro un string*/
250     public static void MenuSel(string tipoMen){
251         //controlamos en menu estamos
252         PreferSctipt.menu = tipoMen;
253     }

```

Fig 178"Función que ayuda a controlar el menu en el que estamos"

```

254  /*Esta función se utiliza para saber que color de botones y panel se esta asignando
255  y de esa manera combinar los colores de manera correcta*/
256  public static void ColB(string col){
257      PreferSctipt.colorBt = col;
258  }
259 }
```

Fig 179"Esta función se usa para saber el color del panel y botones que se está asignando "

#### 1.4.7. Script WebData.cs

Se crea la función *Post()* para consumir el servicio web, pasamos la url del servidor web indicando el tipo de método en este caso POST ya que se va a insertar datos en el servidor web, generamos el archivo json de mis preguntas, envío el json y finalmente hago la petición al servidor web mediante *SendWebRequest()*.

```

using UnityEngine;
using System.Runtime.InteropServices;
using UnityEngine.UI;

public class WebData : MonoBehaviour {

    [DllImport("__Internal")]
    private static extern int GetUrlVal();

    [DllImport("__Internal")]
    private static extern int GetEjer();

    [DllImport('__Internal')]
    private static extern string GetMail();

    public Text idVal;
    public Text idMail;
    public Text idEjer;

    void Start() {
    }

    public int ID(){
        int id = GetUrlVal();
        Debug.Log("ID: "+id);
        idVal.text = " "+id;
        return id;
    }

    public string Mail(){
        string mailV = GetMail();
        Debug.Log("Mail - Un: "+mailV);
        idMail.text = " "+mailV;
        return mailV;
    }

    public int Ejercitario(){
        int ejerV = GetEjer();
        Debug.Log("Ejer - Un: "+ejerV);
        idEjer.text = " "+ejerV;
        return ejerV;
    }
}
```

Fig 180"Script WebData para los servicios web"

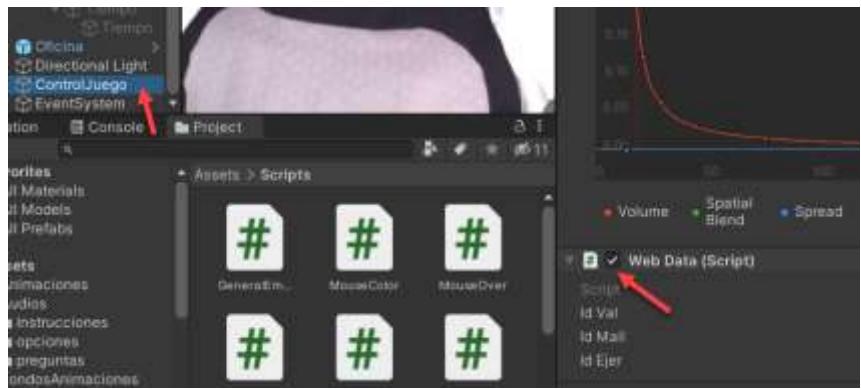


Fig 181 "Añadimos el script WebData al proyecto, lo arrastramos al gameobject ControlJuego"

## 1.5. Programación Simulador Laboral El Tiempo

### 1.5.1. Script GeneralEmpezar.cs

```

using System;
using System.Collections;
using System.Collections.Generic; //libreria para manipular con listas
using UnityEngine;
using System.Linq; //libreria para realizar consultas
using System.Text; //libreria para manipular texto
using UnityEngine.UI; //libreria para acceder a las propiedades de los elementos de la interfaz de usuario.
using System.IDisposable; //libreria para escribir y leer datos
using System.Threading.Tasks;
using UnityEngine.Events;
using TMPro; //Importamos la libreria TextMeshPro;
using Random = UnityEngine.Random;
using UnityEngine.Networking; //Importamos la libreria para la comunicacion con el servidor web.

```

Fig 182 "Librerias utilizadas"

### 1.5.1.1 Declaración de variables y relación con objetos de interfaz de Unity



```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Numerics;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;
using UnityEngine.EventSystems;
using System.Threading.Tasks;
using UIEvent;

public class MainController : IInputHandler
{
    // Se declaran los diccionarios para las imágenes, audios y texturas que se utilizarán.
    public Dictionary<string, Texture2D> texturas = new Dictionary<string, Texture2D>();
    public Dictionary<string, AudioClip> audios = new Dictionary<string, AudioClip>();
    public Dictionary<string, Image> imagenes = new Dictionary<string, Image>();

    public List<Image> lista_imagenes = new List<Image>();
    public List<AudioClip> lista_audios = new List<AudioClip>();

    public string nombre_escena_actual;
    public float volumen;

    public void OnSelect(EventSystem e)
    {
        // Se obtiene el objeto que ha sido seleccionado.
        GameObject go = e.currentSelectedGameObject;
        if (go != null)
        {
            // Se obtiene la componente de evento que se ha seleccionado.
            UIEvent uiEvent = go.GetComponent<UIEvent>();
            if (uiEvent != null)
            {
                // Se obtiene el nombre del objeto que ha sido seleccionado.
                string nombre = uiEvent.nombre;
                // Se obtiene el nombre del componente que se ha seleccionado.
                string nombreComponente = uiEvent.nombreComponente;
                // Se obtiene el nombre del objeto que ha sido seleccionado y su posición en la lista.
                int indice = lista_imagenes.IndexOf(go);
                if (indice > -1)
                {
                    // Se actualiza la imagen que se ha seleccionado.
                    imagenes[nombre] = lista_imagenes[indice];
                    // Se actualiza el nombre del componente que se ha seleccionado.
                    audios[nombre] = lista_audios[indice];
                    // Se actualiza el nombre del objeto que ha sido seleccionado.
                    nombre_escena_actual = nombre;
                }
            }
        }
    }

    public void OnDeselect(EventSystem e)
    {
        // Se obtiene el objeto que ha sido desseleccionado.
        GameObject go = e.currentDesselectedGameObject;
        if (go != null)
        {
            // Se obtiene la componente de evento que se ha desseleccionado.
            UIEvent uiEvent = go.GetComponent<UIEvent>();
            if (uiEvent != null)
            {
                // Se obtiene el nombre del objeto que ha sido desseleccionado.
                string nombre = uiEvent.nombre;
                // Se obtiene el nombre del componente que se ha desseleccionado.
                string nombreComponente = uiEvent.nombreComponente;
                // Se obtiene el nombre del objeto que ha sido desseleccionado y su posición en la lista.
                int indice = lista_imagenes.IndexOf(go);
                if (indice > -1)
                {
                    // Se actualiza la imagen que se ha desseleccionado.
                    imagenes[nombre] = lista_imagenes[indice];
                    // Se actualiza el nombre del componente que se ha desseleccionado.
                    audios[nombre] = lista_audios[indice];
                    // Se actualiza el nombre del objeto que ha sido desseleccionado.
                    nombre_escena_actual = nombre;
                }
            }
        }
    }

    public void Update()
    {
        // Se obtiene el volumen actual.
        volumen = PlayerPrefs.GetFloat("Volumen");
        // Se actualiza el volumen de los audios.
        foreach (var item in audios)
        {
            item.Value.volume = volumen;
        }
    }
}
```

Fig 183 "Declaración de variables parte 1"

Para la navegación por teclado se declararon arreglos seleccionables, arreglos de imágenes y de audios, por tanto, es importante recalcar que el orden en que se cargue los objetos en cada arreglo será el orden que se seleccione al presionar el teclado, es decir si al presionar el teclado se activa el botón en la posición 0 entonces también se activará la imagen y sonido en la posición 0.

```

//Este script es el que se ejecuta en el controlador para el juego principal, cada instrucción
//que devuelva una acción de audio en audio
//sección de la función devolverá los sonidos de los efectos vertebrales a las instrucciones sobre el personaje.
public AudioClip[] sonidoInstruc_preguntasFinal;

int puntoJfinal = 0;
private float StartTime;
//Variables para controlar los niveles de dificultad.
public static string menu = "Main";
//Aqui se definen los diferentes tipos de celdas y sus posiciones en el tablero.
//nxtFieldMain, nxtFieldPrefer, nxtFieldDiag, nxtFieldPreg, nxtFieldPin, nxtFieldFin;
public Selectable[] nxtFieldMain, nxtFieldPrefer, nxtFieldDiag, nxtFieldPreg, nxtFieldPin, nxtFieldFin;

public Image[] main, prefer, diag, joqt, finA, finAT;
//Celdas que tienen que haber en las filas del panel de alternativas.
//nxtCeldas que tienen que haber en las filas del panel de alternativas.
//finCeldas que tienen que haber en las filas del panel de alternativas.
//finAT que tienen que haber en las filas del panel de alternativas.
//finCeldas que tienen que haber en las filas del panel de alternativas.
//finAT que tienen que haber en las filas del panel de alternativas.

int punt = 0, nroI = 0;
//nroI: Contador para saber en que índice del panel de alternativas estamos.
//nroF: Contador para saber en que ímpar del panel de alternativas estamos.
int nroCP = 0, nroFP = 0;
//nroCP: Contador para saber en que posición estamos.
//nroFP: Contador para saber en que ímpar del panel de alternativas estamos.
int nroEF = 0, nroFF = 0;
//nroEF: Contador para saber en que índice del panel de alternativas estamos.
//nroFF: Contador para saber en que índice del panel de alternativas estamos.
int nroTA = 0, nroTPA = 0;
int nroTEstreno = 0;
public AudioSource sonidoAS;

//Variables que sirven para iniciar el juego de un modo controlado por el usuario.
public AudioSource[] audiosMainMouse, audiosPreferMouse, audiosDiagMouse, audiosPregMouse, audiosFinMouse, audiosInMouse;
public AudioSource audiosMain, audiosPrefer, audiosDiag, audiosPreg, audiosFin, audiosIn;
//variables que sirven para iniciar el juego de un modo controlado por el usuario.
public static bool nuevoCamara = true;
//variables que sirven para controlar a la cual se le relaciona la cámara principal de la escena.
public GameObject camara;

```

Fig 184"Declaración de variables parte 2"

```

//Lista de tipo entero en donde se guardara los números aleatorios.
List<int> numerosGuardados;
List<string> jerarqre;
//Lista en donde se guardan las preguntas. //
List<int> numPares;
//Lista de enteros, en donde se guardan las alternativas. //
List<int> listaFinalPregs;
//Variable que obtiene la fecha actual. //
string datetime;
//BANDERA que se usa para indicar que se ha llegado al final del juego, a esta variable la asociamos al botón siguiente. //
public Button sigoFin;
//variables de tipo Sprite.
//ImagenCasa: imagen para reiniciar juego
//ImagenSiguiente: imagen para botón siguiente
//
public Sprite ImagenCasa, ImagenSiguiente;
int conteoAlter = 0, audiosConteo = 0;
//CREACION DE UNA VARIABLE DE LA CLASE WebData,
//la cual se pasa los campos de datos para consumir los servicios web.
longo relacionando esta variable con el gameobject que contiene el script WebData.cs. //
public WebData webdata;
//Instancia de la clase SimuladorData //
SimuladorData ejercicio = new SimuladorData();
//variable de la clase Pregunta, esta variable irá agregando dentro del JSON una nueva pregunta//
Pregunta ppre;
string rotFin = "";
public TMP_InputField[] jerarquiaInput;
float timePress = 0f;

```

Fig 185"Declaración de variables parte 3"

Desde la ventana de jerarquía en el gameobject ControlJuego una vez posicionados aquí nos dirigimos a la ventana inspector y procedemos a relacionar los objetos con cada variable.

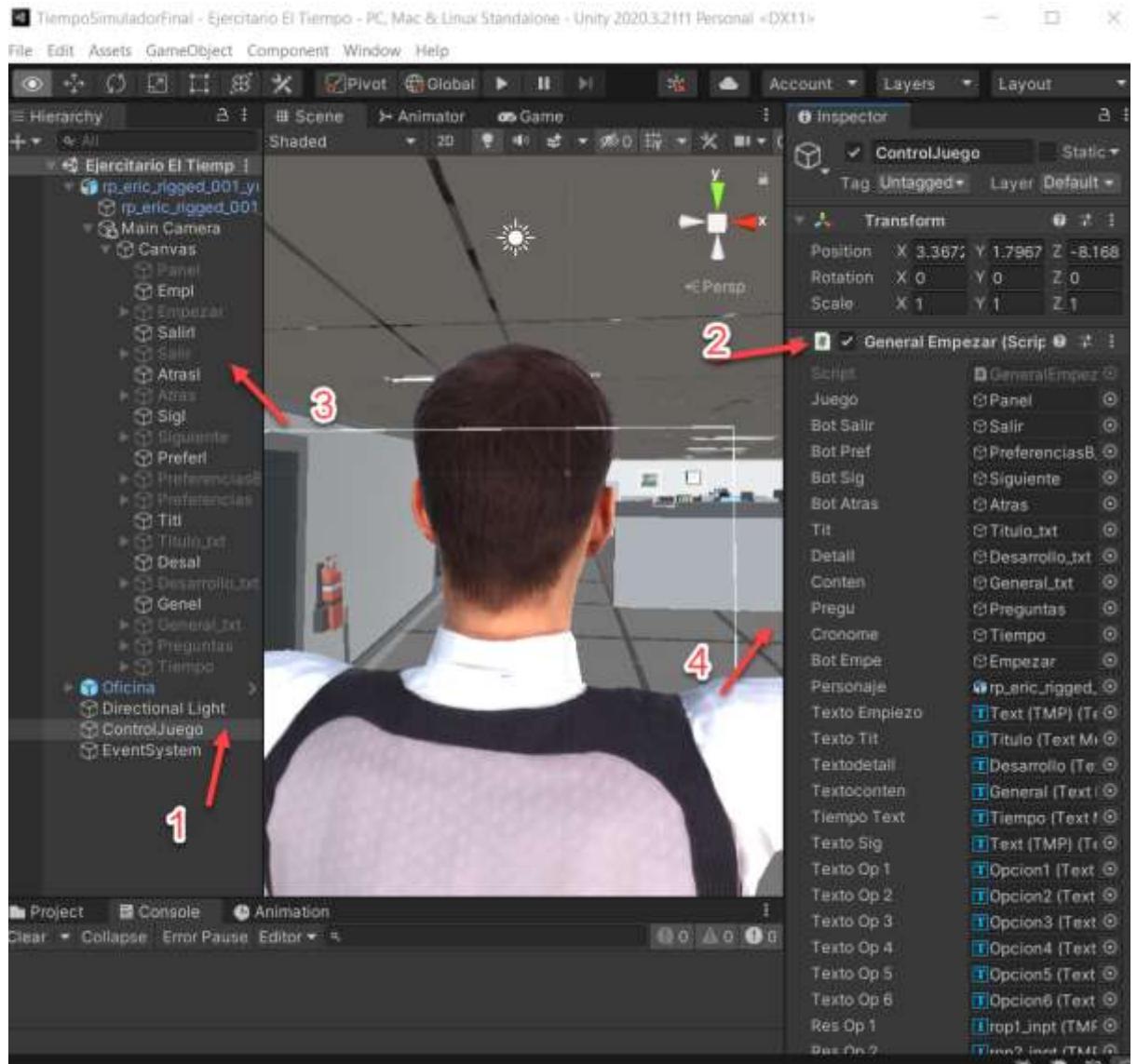


Fig 186 "Relacionamos cada gamobject de la escena con la variable determinada"

### 1.5.1.2 Declaración de funciones

```

void Start()
{
    //Llamadas a la función para que actualice los valores iniciales de la
    nuevoJuego();
}

void Update()
{
    //Si no se juega a Preguntas
    if (GeneralEmpezar.menu == "Preguntas")
    {
        corriendo();
        //Comprueba si toca el tiempo
    }
    //Si el botón comienza este acción y el movimiento de la cámara esta activa,
    entonces se activa la cámara con el mouse
    if (botEmp.activeSelf == true && mouseCamera == true)
    {
        float pointer_x = Input.GetAxis("Mouse X");
        float pointer_y = Input.GetAxis("Mouse Y");
        camera.transform.Rotate(0, pointer_x * 1f, 0);
    }
    if (botEmp.activeSelf == true) //Si el botón comienza esta acción
    {
        //Se activa el personaje
        personaje.SetActive(true);
    }
    //Comprueba si la tecla Tabulando, para que siempre este buscando a los eventos del teclado
    Tabulando();

    //Si el botón que comienza cuando las imágenes de respuesta
    if (GeneralEmpezar.menu == "Preguntas" && (resOp1.text != "" && resOp2.text != "" && resOp3.text != "" &&
    resOp5.text != "" && resOp6.text != "" && diasOp1.text != "" && diasOp2.text != "" && diasOp3.text != "" &&
    diasOp4.text != "" && diasOp5.text != "" && diasOp6.text != ""))
    {
        //Se activa el botón siguiente
        botSig.SetActive(true);
    }
    //Si alguno de los campos de respuesta tienen texto vacío o uno más de los activados el botón siguiente
    else if (GeneralEmpezar.menu == "Preguntas" && ((resOp1.text == "" && resOp2.text == "") ||
    resOp3.text == "" && resOp4.text == "" && resOp5.text == "" && resOp6.text == "") ||
    diasOp1.text == "" && diasOp2.text == "" && diasOp3.text == "" && diasOp4.text == "") ||
    diasOp5.text == "" && diasOp6.text == "") || (resOp1.text == null || resOp2.text == null || resOp3.text == null ||
    resOp4.text == null || resOp5.text == null || resOp6.text == null) || (diasOp1.text == null || diasOp2.text == null || diasOp3.text == null || diasOp4.text == null ||
    diasOp5.text == null || diasOp6.text == null)) {
        botSig.SetActive(false); //Se desactiva el botón siguiente
    }
    if (timePress > 0f) //Si la cantidad de tiempo de presión de una determinada tecla es mayor a 0f
    timePress -= Time.deltaTime; //se reduce el tiempo
}
//Si presiono la tecla Tabulando, revisa si en este momento, tabulando es la tecla que se está calculando de tiempo en dirección de la tecla en menor a igual a 0
if (((Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow) || Input.GetKeyDown(KeyCode.DownArrow) ||
Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow)) || Input.GetKeyDown(KeyCode.LeftAlt)) && timePress <= 0) {
    //Cambia la tecla que se está calculando de tiempo en dirección de la tecla menor
    timePress = 0f; //Vuelve que la cantidad de tiempo de presión de la tecla sea 0f
}

```

Fig 187"Funciones Start() y Update()"

```

/*Función que se llamará desde la función Siguiente() o la función Atas() reinicia sus valores a cero*/
public void ReinicioSeleccion()
{
    /*Inicializo todos los contadores en cero*/
    numC = 0;
    numI = 0;
    numCP = 0;
    numFP = 0;
    numPP = 0;
    numPPP = 0;
    numFF = 0;
    numIFF = 0;
    numFFA = 0;
    numIFFA = 0;

    nextFieldDiag[numC].Select(); //selecciona el objeto que se encuentra en la posición
    nextFieldPrefer[numCP].Select(); //selecciona el objeto que se encuentra en la posición
    nextFieldPreg[numPP].Select(); //selecciona el objeto que se encuentra en la posición
    nextFieldFin[numFF].Select(); //selecciona el objeto que se encuentra en la posición
    nextFieldFinA[numFFA].Select(); //selecciona el objeto que se encuentra en la posición

    /*modo a desactivar todas las imágenes los paneles*/
    foreach (var item in prefImg)
    {
        item.enabled = false;
    }
    foreach (var item in dial)
    {
        item.enabled = false;
    }
    foreach (var item in juegl)
    {
        item.enabled = false;
    }
    foreach (var item in finI)
    {
        item.enabled = false;
    }
    foreach (var item in finAI)
    {
        item.enabled = false;
    }
}

```

Fig 188"Función ReinicioSelección()"

```

247     public void ReinicioTiempo()
248     {
249         StartTime = Time.time; //inicia el tiempo
250     }
251

```

Fig 189"Función ReinicioTiempo()"

```

/*Esta función permite salir del simulador laboral*/
public void Salir()
{
    Application.Quit();
}

```

Fig 190"Función Salir()"

```

private void nuevoJuego()
{
    //Iniciamos el conteo de las variables
    jeraTgrie = new List<string>();
    //Asignamos el tiempo de inicio cuando los valores no son iguales entre los dos segundos (00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60)
    ejercicio.setTiempoInicio(System.DateTime.Now.Hour.ToString("00") + ":" + System.DateTime.Now.Minute.ToString("00") + ":" + System.DateTime.Now.Second.ToString("00"));
    controlInstruc = 0;
    controlMater = 0;
    animarCante = 0;
    //Contador de segundos - 20s
    restoP1.text = "";
    restoP2.text = "";
    restoP3.text = "";
    restoP4.text = "";
    restoP5.text = "";
    restoP6.text = "";
    diaSop1.text = "";
    diaSop2.text = "";
    diaSop3.text = "";
    diaSop4.text = "";
    diaSop5.text = "";
    diaSop6.text = "";
    //Creando una nueva lista de un bucle de 10, en donde los presentes si van para las respuestas de
    //jerceratigre y las otras 10 para las respuestas de alternativas de cada pregunta
    respuestasadas = new List<string> { "0", "0", "0", "0", "0", "0", "0", "0", "0", "0" };
    //Rediseña la interfaz de usuario para la variable alternativas en instrucciones
    //dado que se usan en instrucciones
    instrucciones = new List<string>(instrucGene.Split(new string[] { ";" }, StringSplitOptions.None));
    //Algunas veces se presentan errores en la ejecución al intentar inicializar todas las variables en el
    ejercicio.ReiniciaPreguntas();
}

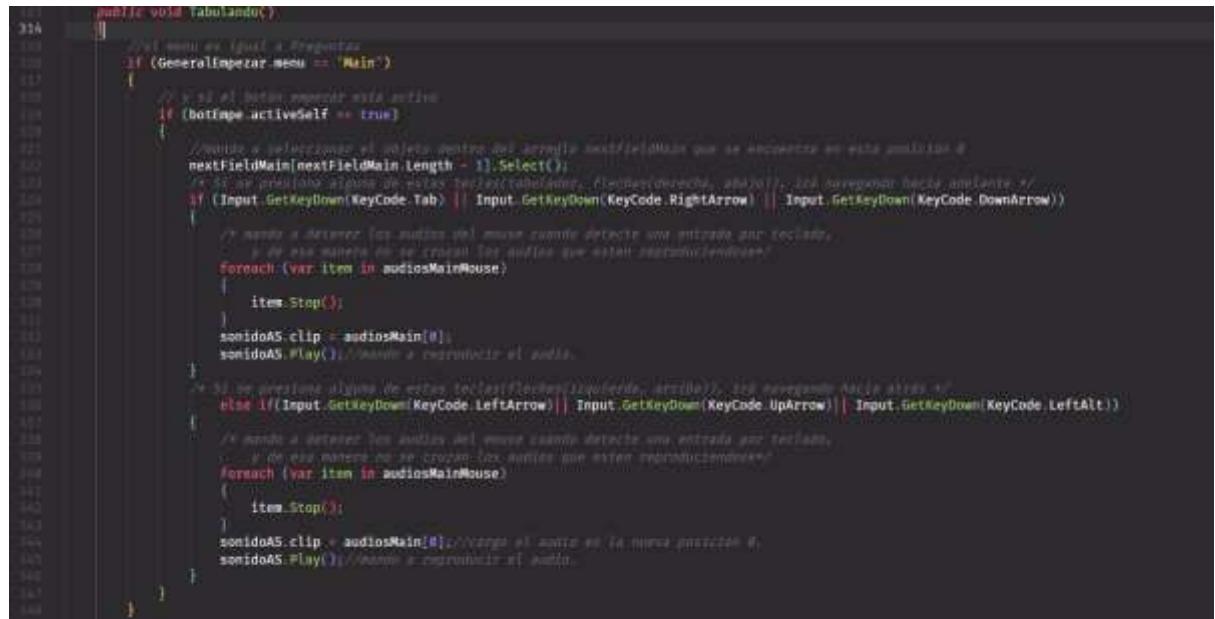
try
{
    //Inicializa las listas
    conteneProg = new List<string>();
    alternativas = new List<string>();
    alternativasBias = new List<string>();
    linea = documentos[0].split("\n");
    //Guarda en la variable linea la primera linea del contenido de instrucciones
    //y linea es diferente de null es decir que esta linea
    //no es la introducción de los comienzos de lectura
    if (linea != null)
    {
        //Guarda linea linea en donde se separan los datos de contenido de instrucciones, donde cada uno que empieza con ; y lo guarda en conteneProgFinal
        conteneProgFinal = new List<string>(linea.Split(new string[] { ";" }, StringSplitOptions.None));
        //Guarda linea linea en donde se separan los datos de contenido de instrucciones, donde cada uno que empieza con ; y lo guarda en alternativasFinal
        alternativasFinal = new List<string>(linea.Split(new string[] { ";" }, StringSplitOptions.None));
        //Guarda linea linea en donde se separan los datos de contenido de instrucciones, donde cada uno que empieza con ; y lo guarda en alternativasBiasFinal
        alternativasBiasFinal = new List<string>(linea.Split(new string[] { ";" }, StringSplitOptions.None));
    }
}
catch (Exception e)
{
    Debug.Log("Exception: " + e.Message);
}

```

Fig 191 "Función nuevoJuego()"

La función *Tabulando()* controla los eventos del teclado, es decir la navegación en la interfaz al presionar ciertas teclas, para lograrlo primero se declaran arreglos de tipo Selectable, luego asociamos con cada gameobject de la escena, luego agregamos a nuestra sentencia de código el método *Select()* de Unity el cual hace que se seleccione el componente requerido cada vez que se navegue con el teclado; para controlar en que parte del simulador se encuentra el participante se hace uso de la variable estática *menu*, ésta variable va cambiando de nombre en cada panel que nos encontramos, por ejemplo para indicar que estamos en el panel principal decimos que menu es igual a *Main*, cuando estamos en el panel de instrucciones menu es igual a *Jugando*, cuando pasemos al panel

de preguntas menu es igual a *Preguntas*, cuando pasemos al panel de calificación menu es igual a *FinA* y cuando pasemos al último panel menu es igual a *Fin*.



```
117     public void Tabulando()
118     {
119         if (menu == Input.Menu.Preguntas)
120         {
121             if (GeneralImprimir.menu == "Main")
122             {
123                 // si el botón repeat está activo
124                 if (BotonRepeat.activeSelf == true)
125                 {
126                     // Selecciona el objeto dentro del array de nextFieldMain que se encuentra en esta posición. Es
127                     nextFieldMain[nextFieldMain.Length - 1].Select();
128                     // Selecciona el objeto de estos nextFieldTabulado, Flechaizquierda, abajo, flechaderecha, arriba
129                     if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
130                     {
131                         // para a detener los audios del mouse cuando detecte una entrada por teclado,
132                         // y de esa manera no se crean los audios que están sincronizados
133                         foreach (var item in audiosMainMouse)
134                         {
135                             item.Stop();
136                         }
137                         sonidoA5.clip = audiosMain[0];
138                         sonidoA5.Play(); //carga el sonido en la nueva posición 0;
139                     }
140                 }
141             }
142         }
143     }
144 }
```

Fig 192"Control por teclado función Tabulando() parte 1"

```

118 //Control de los botones del panel de instrucciones
119 if (GeneralEmpezar.menu == "Jugando")
120 {
121     //Iniciar panel de instrucciones
122     if (empiezo.enabled == false)
123     {
124         //Mover a desactivar todos los objetos del panel iniciar
125         foreach (var item in main)
126         {
127             item.enabled = false;
128         }
129         //El comando solo ejecuta el comando del arreglo nextFieldDiag
130         if (numC < nextFieldDiag.length - 1)
131         {
132             if (numC == 0) //Si numC es igual a 0 quiere decir que estoy seleccionando el botón que está en la posición 0
133                 //Mando a seleccionar el objeto dentro del arreglo nextFieldDiag que se encuentra en la posición actual
134                 nextFieldDiag[numC].Select();
135                 //Activo la imagen del botón que está en la posición actual
136                 dial[numC].enabled = true;
137
138             //Si se presiona alguno de estos teclados (ALT, Flechasderecha, TAB), iré a seguir la fila adelante
139             if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
140
141                 //Monto a desactivar los botones del mouse cuando detecta una entrada por teclado,
142                 //y de esa manera no se activan los botones que están reproduciéndose
143                 foreach (var item in audiosDiagMouse)
144                 {
145                     item.Stop();
146                 }
147             //Si numC es igual a 0 quiere decir que estoy seleccionando el botón que está en la posición 0
148             if (numC == 0)
149             {
150                 sonidoAS.Stop(); //Desengo todos los audios
151                 sonidoAS.clip = audioDiag[0]; //Cargo el audio en la posición 0
152                 sonidoAS.Play(); //Mando a reproducir el audio
153
154             //Si el comando no igual a 0
155             if (numC != 0)
156             {
157                 num++; //Incremento en 1 la posición de mi botón
158                 numI++; //Incremento en 1 la posición a la imagen de mi botón
159                 nextFieldDiag[numC].Select(); //Mando a seleccionar el botón en la nueva posición
160                 sonidoAS.Stop(); //Desengo todo
161                 sonidoAS.clip = sonidoInstruc[conteoInstruc]; //Cargo el audio de instrucciones en la posición actual de conteoInstruc
162                 sonidoAS.Play(); //Mando a reproducir el audio
163
164             }
165         }
166     }
167 }
168 //Caso para cuando
169
170 numC++; //Incremento en 1 a la posición de mi botón
171 numI++; //Incremento en 1 a la posición a la imagen de mi botón
172 nextFieldDiag[numC].Select(); //Mando a seleccionar el botón en la nueva posición
173 sonidoAS.Stop();
174 //Cargo el audio de instrucciones en la posición actual de numC
175 sonidoAS.clip = audioDiag[numC];
176 sonidoAS.Play(); //Mando a reproducir el audio
177
178 //Activar la imagen del botón de la posición anterior
179 dial[numI - 1].enabled = false;
180 //Activar la imagen del botón en la nueva posición
181 dial[numI].enabled = true;

```

Fig 193 "Control por teclado función Tabulando parte 2"

```

413     if (Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow) || Input.GetKeyDown(KeyCode.LeftAlt))
414     {
415         // si se presiona alguna de estas teclas(tabulador, flechas(derecha, abajo)), irá navegando hacia adelante
416         // y de esa manera no se cruzan los audios que estén reproduciéndose
417         foreach (var item in audiosDiagMouse)
418         {
419             item.Stop();
420         }
421         numC++;
422         // si el tamaño del contador numC es igual al tamaño del arreglo nextFieldDiag
423         if (numC == nextFieldDiag.Length - 1)
424         {
425             // mando a detener los audios del mouse cuando detecte una entrada por teclado,
426             // y de esa manera no se cruzan los audios que estén reproduciéndose
427             foreach (var item in audiosDiagMouse)
428             {
429                 item.Stop();
430             }
431             numC = nextFieldDiag.Length - 1;
432             nextFieldDiag[numC].Select(); //selecciono el botón en la posición actual
433             sonidoAS.Stop(); //detengo todos los audios que estén reproduciéndose
434             sonidoAS.clip = audiosDiag[numC]; //carga el audio en la nueva posición
435             sonidoAS.Play(); //mando a reproducir el audio
436             dial[numI].enabled = true; //activa la imagen del botón en la posición actual
437             Debug.Log("++" + numC + " < " + (nextFieldDiag.Length - 1));
438         }
439         // si el tamaño del contador numC es igual al tamaño del arreglo
440         else if (numC == 5)
441         {
442             numC++; //incremento en 1 a la posición del contador
443             numI++; //incremento en 1 a la posición del contador
444             // mando a seleccionar el botón en la nueva posición
445             nextFieldDiag[numC].Select();
446             sonidoAS.Stop();
447             sonidoAS.clip = sonidoInstru.cantosInstru; //carga el audio en la posición actual de cantos instru
448             sonidoAS.Play();
449             dial[numI].enabled = false; //desactiva la imagen del botón en la posición actual
450             dial[numI].enabled = true; //activa la imagen del botón en la posición actual
451             Debug.Log("++" + numC);
452         }
453         // si el tamaño del contador numC es igual al tamaño del arreglo
454         else if (numC == nextFieldDiag.Length - 1)
455         {
456             numC++; //incremento en 1 a la posición del contador
457             numI++; //incremento en 1 a la posición del contador
458             // mando a seleccionar el botón en la nueva posición
459             nextFieldDiag[numC].Select();
460             sonidoAS.Stop();
461             // mando a detener los audios del mouse cuando detecte una entrada por teclado,
462             // y de esa manera no se cruzan los audios que estén reproduciéndose
463             foreach (var item in audiosDiagMouse)
464             {
465                 item.Stop();
466             }
467             sonidoAS.clip = audiosDiag[numC];
468             sonidoAS.Play();
469             dial[numI].enabled = false; //desactiva la imagen del botón en la posición actual
470             dial[numI].enabled = true; //activa la imagen del botón en la posición actual
471             Debug.Log("++" + numC);
472         }
473     }

```

Fig 194"Control por teclado función Tabulando() parte 3"

```

457     //Si el tamaño del contador numC es igual al tamaño del arreglo nextFieldDiag
458     else if (numC == nextFieldDiag.Length - 1)
459     {
460         //+ Si se presiona alguna de estas teclas(tabulador, flechas(derecha, abajo)), irá navegando hacia adelante */
461         if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
462         {
463             // mando a detener los audios del mouse cuando detecte una entrada por teclado,
464             // y de esa manera no se cruzan los audios que estén reproduciéndose
465             foreach (var item in audiosDiagMouse)
466             {
467                 item.Stop();
468             }
469             dial[numI].enabled = false; //entonces se desactiva el objeto que está en esa posición.
470             numC = 0; //se inicia todo en 0
471             numI = 0; //se inicia todo en 0
472             nextFieldDiag[numC].Select(); //selecciono el botón en la posición actual.
473             sonidoAS.Stop(); //detengo todos los audios.
474             sonidoAS.clip = audiosDiag[numC]; //agrego el audio en la nueva posición.
475             sonidoAS.Play(); //mando a reproducir el audio
476             dial[numI].enabled = true; //mando a activar la imagen del botón en la posición actual.
477         }
478         //+ Si se presiona alguna de estas teclas(flechas(izquierda, arriba)), irá navegando hacia atrás */
479         else if (Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow) || Input.GetKeyDown(KeyCode.LeftAlt))
480         {
481             numC--; //decremento en 1 a la posición del contador.
482             numI--; //decremento en 1 a la posición del contador.
483             // mando a detener los audios del mouse cuando detecte una entrada por teclado,
484             // y de esa manera no se cruzan los audios que estén reproduciéndose
485             foreach (var item in audiosDiagMouse)
486             {
487                 item.Stop();
488             }
489             dial[numI + 1].enabled = false; //desactivo la imagen del botón de la posición actual mas 1.
490             nextFieldDiag[numC].Select(); //selecciono el botón en la posición actual.
491             sonidoAS.Stop(); //detengo todos los audios que estén reproduciéndose.
492             sonidoAS.clip = audiosDiag[numC]; //carga el audio en la nueva posición.
493             sonidoAS.Play(); //mando a reproducir el audio.
494             dial[numI].enabled = true; //activo la imagen del botón de la posición actual.
495         }
496     }

```

Fig 195"Control por teclado función Tabulando() parte 4"

```

19 // si el usuario ha presionado alguna tecla que elige el modo de pregunta
20 if (pregu.activeSelf == true)
21 {
22     // este botón sirve para activar o desactivar las preguntas
23     GeneraTempear.MenuSel("Preguntas");
24     nextFieldDingding.Select();
25     // numero de respuesta que se muestra en la pantalla actual
26     foreach (var item in dia)
27     {
28         item.enabled = false;
29     }
30     numC = 0; // inicializar el contenido en 0
31     numI = 0; // inicializar el contenido en 0
32
33 }
34
35 // modo de igual a preguntas
36 else if (GeneraTempear.menu == "Preguntas")
37 {
38     // aumentar en 10 el conteo de número de posiciones del arreglo nextFieldDingding
39     int totPreg = 0;
40     // los primeros 10 de preguntas están activos
41     if (pregu.activeSelf == true)
42     {
43         // si el botón aparece visto activo
44         if (botSig.activeSelf == true)
45         {
46             // se suma al valor del botón del arreglo de 10 a 10 más
47             totPreg = nextFieldPreg.Length + 10;
48         }
49         // el botón siguiente visto desactivado
50     else if (botSig.activeSelf == false)
51     {
52         // se resta el valor del botón del arreglo de 10 a 10
53         totPreg = nextFieldPreg.Length - 10;
54     }
55 }
56
57 // igual a igual sigue
58 nextSig.enabled = true;

```

Fig 196"Control por teclado función Tabulando() parte 5"

```

59
60 // si el usuario ha presionado tabuleo
61 if (numPP < totPreg)
62 {
63     // si se presiona alguno de estos teclas tabuleo, flecha izquierda, flecha derecha, flecha abajo
64     if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
65     {
66         // recorremos cada elemento de la lista de preguntas
67         // la variable item es el modo de actividad
68         foreach (var item in preguntas)
69         {
70             item.interactable = true;
71         }
72         // si el conteo es menor a 0
73         if (conteoAlter < 0)
74         {
75             // recorremos los contenidos de la lista de audios de las preguntas
76             // si es menor a 0, iniciálos a continuación
77             if (conteoAlter > preguntasSonidos.Length - 1)
78             {
79                 conteoAlter = 0;
80             }
81
82             // recorremos cada elemento de la lista de audios del modo mediante
83             // la variable item y los manda a detener
84             foreach (var item in audiosPreg)
85             {
86                 item.Stop();
87             }
88             // si el conteo de numPP es 0
89             if (numPP == 0)
90             {
91                 nextFieldPreg[numPP].Select(); // selecciona el botón en la posición actual
92                 juegIT(numPP).enabled = true; // cambia la imagen del botón en la posición actual
93                 sonidoAS.Stop();
94                 sonidoAS.clip = audiosPreg[0]; // carga el audio en la nueva posición 0
95                 sonidoAS.Play();
96             }
97             // si el conteo es menor a 2 o mayor a 22
98             if (numPP < 2 || numPP > 22)
99             {
100                 numPP++; // aumenta en 1 la posición de mi botón
101                 juegIT(numPP).enabled = true; // cambia la imagen de mi botón
102                 nextFieldPreg[numPP].Select(); // selecciona el botón en la posición actual
103                 sonidoAS.Stop();
104             }
105         }
106     }
107 }

```

Fig 197"Control por teclado función Tabulando() parte 6"

```

395
396
397 //si el nombre del botón en la posición del contador numPP es igual a op1_act
398 if (nextFieldPreg[numPP].name == "Op1_act")
399 {
400     conteoAlter++;//incremento el contador en 1.
401
402     //recorro cada elemento de la lista inputs varios mediante
403     //la variable item y los mando a bloquear
404     foreach (var item in inputsVarios)
405     {
406         item.interactable = false;
407     }
408     //cargo el audio del arreglo numerosGuardados en la posición 0.
409     sonidoAS.clip = preguntasSonidos[numerosGuardados[0]];
410     sonidoAS.Play(); //música de retroalimentación al usuario
411 }
412
413 //si el nombre del botón en la posición del contador numPP es igual a op2_act
414 if (nextFieldPreg[numPP].name == "Op2_act")
415 {
416     conteoAlter++;
417     //recorro cada elemento de la lista inputs varios mediante
418     //la variable item y los mando a bloquear
419     foreach (var item in inputsVarios)
420     {
421         item.interactable = false;
422     }
423     //cargo el audio del arreglo numerosGuardados en la posición 1.
424     sonidoAS.clip = preguntasSonidos[numerosGuardados[1]];
425     sonidoAS.Play();
426 }
427
428 //si el nombre del botón en la posición del contador numPP es igual a op3_act
429 if (nextFieldPreg[numPP].name == "Op3_act")
430 {
431     conteoAlter++;
432     //recorro cada elemento de la lista inputs varios mediante
433     //la variable item y los mando a bloquear
434     foreach (var item in inputsVarios)
435     {
436         item.interactable = false;
437     }
438     //cargo el audio del arreglo numerosGuardados en la posición 2.
439     sonidoAS.clip = preguntasSonidos[numerosGuardados[2]];
440     sonidoAS.Play();
441 }
442
443 //si el nombre del botón en la posición del contador numPP es igual a op4_act
444 if (nextFieldPreg[numPP].name == "Op4_act")
445 {
446     conteoAlter++;
447     //recorro cada elemento de la lista inputs varios mediante
448     //la variable item y los mando a bloquear
449     foreach (var item in inputsVarios)
450     {
451         item.interactable = false;
452     }
453     //cargo el audio del arreglo numerosGuardados en la posición 3.
454     sonidoAS.clip = preguntasSonidos[numerosGuardados[3]];
455     sonidoAS.Play();
456 }

```

Fig 198"Control por teclado función Tabulando() parte 7"

```

148 // si el nombre del botón en la posición del contenedor numP es igual a 'Op5_act'
149 if (nextFieldProg[numPP].name == "Op5_act")
150 {
151     conteoAlter++;
152     //recorro cada elemento de la lista inputsVariables mediante
153     //la variable item y los monto a blancos
154     foreach (var item in inputsVariables)
155     {
156         item.interactable = false;
157     }
158     //carga el sonido del arreglo numerosGuardados en la posición 4
159     sonidoAS.clip = preguntasSonidos[numerosGuardados[4]];
160     sonidoAS.Play();
161 }
162 // si el nombre del botón en la posición del contenedor numP es igual a 'Op6_act'
163 if (nextFieldProg[numPP].name == "Op6_act")
164 {
165     conteoAlter++;
166     //recorro cada elemento de la lista inputsVariables mediante
167     //la variable item y los monto a blancos
168     foreach (var item in inputsVariables)
169     {
170         item.interactable = false;
171     }
172     //carga el sonido del arreglo numerosGuardados en la posición 5
173     sonidoAS.clip = preguntasSonidos[numerosGuardados[5]];
174     sonidoAS.Play();
175 }
176 // si el nombre del botón en la posición del contenedor numP es igual a 'Op1_inpt'
177 if (nextFieldProg[numPP].name == "Op1_inpt" || nextFieldProg[numPP].name == "rop1_inpt")
178 {
179     conteoAlter++;
180     //recorro cada elemento de la lista inputsVariables mediante
181     //la variable item y los monto a blancos
182     foreach (var item in inputsVariables)
183     {
184         item.interactable = false;
185     }
186     //carga el sonido del arreglo numerosGuardados en la posición 0
187     sonidoAS.clip = preguntasSonidos[numerosGuardados[0]];
188     sonidoAS.Play();
189 }
190 // si el nombre del botón en la posición del contenedor numP es igual a 'Op2_inpt'
191 if (nextFieldProg[numPP].name == "Op2_inpt" || nextFieldProg[numPP].name == "rop2_inpt")
192 {
193     conteoAlter++;
194     //recorro cada elemento de la lista inputsVariables mediante
195     //la variable item y los monto a blancos
196     foreach (var item in inputsVariables)
197     {
198         item.interactable = false;
199     }
200     //carga el sonido del arreglo numerosGuardados en la posición 1
201     sonidoAS.clip = preguntasSonidos[numerosGuardados[1]];
202     sonidoAS.Play();
203 }
204 // si el nombre del botón en la posición del contenedor numP es igual a 'Op3_inpt'
205 if (nextFieldProg[numPP].name == "Op3_inpt" || nextFieldProg[numPP].name == "rop3_inpt")
206 {
207     conteoAlter++;
208     //recorro cada elemento de la lista inputsVariables mediante
209     //la variable item y los monto a blancos
210     foreach (var item in inputsVariables)
211     {
212         item.interactable = false;
213     }
214     //carga el sonido del arreglo numerosGuardados en la posición 2
215     sonidoAS.clip = preguntasSonidos[numerosGuardados[2]];
216     sonidoAS.Play();
217 }
218 // si el nombre del botón en la posición del contenedor numP es igual a 'Op4_inpt'
219 if (nextFieldProg[numPP].name == "Op4_inpt" || nextFieldProg[numPP].name == "rop4_inpt")
220 {
221     conteoAlter++;
222     //recorro cada elemento de la lista inputsVariables mediante
223     //la variable item y los monto a blancos
224     foreach (var item in inputsVariables)
225     {
226         item.interactable = false;
227     }
228     //carga el sonido del arreglo numerosGuardados en la posición 3
229     sonidoAS.clip = preguntasSonidos[numerosGuardados[3]];
230     sonidoAS.Play();
231 }
232 // si el nombre del botón en la posición del contenedor numP es igual a 'Op5_inpt'
233 if (nextFieldProg[numPP].name == "Op5_inpt" || nextFieldProg[numPP].name == "rop5_inpt")
234 {
235     conteoAlter++;
236     //recorro cada elemento de la lista inputsVariables mediante
237     //la variable item y los monto a blancos
238     foreach (var item in inputsVariables)
239     {
239         item.interactable = false;
240     }
241     //carga el sonido del arreglo numerosGuardados en la posición 4
242     sonidoAS.clip = preguntasSonidos[numerosGuardados[4]];
243     sonidoAS.Play();
244 }
245 // si el nombre del botón en la posición del contenedor numP es igual a 'Op6_inpt'
246 if (nextFieldProg[numPP].name == "Op6_inpt" || nextFieldProg[numPP].name == "rop6_inpt")
247 {
248     conteoAlter++;
249     //recorro cada elemento de la lista inputsVariables mediante
250     //la variable item y los monto a blancos
251     foreach (var item in inputsVariables)
252     {
253         item.interactable = false;
254     }
255     //carga el sonido del arreglo numerosGuardados en la posición 5
256     sonidoAS.clip = preguntasSonidos[numerosGuardados[5]];
257     sonidoAS.Play();
258 }
259 // si el nombre del botón en la posición del contenedor numP es igual a 'rop1_inpt'
260 if (nextFieldProg[numPP].name == "rop1_inpt")
261 {
262     conteoAlter++;
263     //recorro cada elemento de la lista inputsVariables mediante
264     //la variable item y los monto a blancos
265     foreach (var item in inputsVariables)
266     {
267         item.interactable = false;
268     }
269     //carga el sonido del arreglo numerosGuardados en la posición 0
270     sonidoAS.clip = preguntasSonidos[numerosGuardados[0]];
271     sonidoAS.Play();
272 }
273 // si el nombre del botón en la posición del contenedor numP es igual a 'rop2_inpt'
274 if (nextFieldProg[numPP].name == "rop2_inpt")
275 {
276     conteoAlter++;
277     //recorro cada elemento de la lista inputsVariables mediante
278     //la variable item y los monto a blancos
279     foreach (var item in inputsVariables)
280     {
281         item.interactable = false;
282     }
283     //carga el sonido del arreglo numerosGuardados en la posición 1
284     sonidoAS.clip = preguntasSonidos[numerosGuardados[1]];
285     sonidoAS.Play();
286 }
287 // si el nombre del botón en la posición del contenedor numP es igual a 'rop3_inpt'
288 if (nextFieldProg[numPP].name == "rop3_inpt")
289 {
290     conteoAlter++;
291     //recorro cada elemento de la lista inputsVariables mediante
292     //la variable item y los monto a blancos
293     foreach (var item in inputsVariables)
294     {
295         item.interactable = false;
296     }
297     //carga el sonido del arreglo numerosGuardados en la posición 2
298     sonidoAS.clip = preguntasSonidos[numerosGuardados[2]];
299     sonidoAS.Play();
300 }
301 // si el nombre del botón en la posición del contenedor numP es igual a 'rop4_inpt'
302 if (nextFieldProg[numPP].name == "rop4_inpt")
303 {
304     conteoAlter++;
305     //recorro cada elemento de la lista inputsVariables mediante
306     //la variable item y los monto a blancos
307     foreach (var item in inputsVariables)
308     {
309         item.interactable = false;
310     }
311     //carga el sonido del arreglo numerosGuardados en la posición 3
312     sonidoAS.clip = preguntasSonidos[numerosGuardados[3]];
313     sonidoAS.Play();
314 }
315 // si el nombre del botón en la posición del contenedor numP es igual a 'rop5_inpt'
316 if (nextFieldProg[numPP].name == "rop5_inpt")
317 {
318     conteoAlter++;
319     //recorro cada elemento de la lista inputsVariables mediante
320     //la variable item y los monto a blancos
321     foreach (var item in inputsVariables)
322     {
323         item.interactable = false;
324     }
325     //carga el sonido del arreglo numerosGuardados en la posición 4
326     sonidoAS.clip = preguntasSonidos[numerosGuardados[4]];
327     sonidoAS.Play();
328 }
329 // si el nombre del botón en la posición del contenedor numP es igual a 'rop6_inpt'
330 if (nextFieldProg[numPP].name == "rop6_inpt")
331 {
332     conteoAlter++;
333     //recorro cada elemento de la lista inputsVariables mediante
334     //la variable item y los monto a blancos
335     foreach (var item in inputsVariables)
336     {
337         item.interactable = false;
338     }
339     //carga el sonido del arreglo numerosGuardados en la posición 5
340     sonidoAS.clip = preguntasSonidos[numerosGuardados[5]];
341     sonidoAS.Play();
342 }
343 // si el nombre del botón en la posición del contenedor numP es igual a 'titular.txt'
344 if (nextFieldProg[numPP].name == "titular.txt")
345 {
346     conteoAlter++;
347     //recorro cada elemento de la lista inputsVariables mediante
348     //la variable item y los monto a blancos
349     foreach (var item in inputsVariables)
350     {
351         item.interactable = false;
352     }
353     //carga el sonido del arreglo numerosGuardados en la posición 0
354     sonidoAS.clip = preguntasSonidos[numerosGuardados[0]];
355     sonidoAS.Play();
356 }
357 // si el nombre del botón en la posición del contenedor numP es igual a 'desarrolla.txt'
358 if (nextFieldProg[numPP].name == "desarrolla.txt")
359 {
360     conteoAlter++;
361     //recorro cada elemento de la lista inputsVariables mediante
362     //la variable item y los monto a blancos
363     foreach (var item in inputsVariables)
364     {
365         item.interactable = false;
366     }
367     //carga el sonido del arreglo numerosGuardados en la posición 1
368     sonidoAS.clip = preguntasSonidos[numerosGuardados[1]];
369     sonidoAS.Play();
370 }
371 }

```

Fig 199"Control por teclado función Tabulando() parte 8"

```

698 // si el nombre del botón en la posición del contenedor numP es igual a 'PreferenciasB.txt'
699 if (nextFieldProg[numPP].name == "PreferenciasB.txt")
700 {
701     conteoAlter++;
702     //recorro cada elemento de la lista inputsVariables mediante
703     //la variable item y los monto a blancos
704     foreach (var item in inputsVariables)
705     {
706         item.interactable = false;
707     }
708     //carga el sonido del arreglo numerosGuardados en la posición actual
709     sonidoAS.Stop();
710     //carga el sonido del arreglo numerosGuardados en la posición actual
711     nextFieldProg[numPP].Select(); //selecciona el botón en la posición actual
712     sonidoAS.Play();
713 }
714 // si el nombre del botón en la posición del contenedor numP es igual a 'PreferenciasB.txt'
715 if (nextFieldProg[numPP].name == "PreferenciasB.txt")
716 {
717     conteoAlter++;
718     //recorro cada elemento de la lista inputsVariables mediante
719     //la variable item y los monto a blancos
720     foreach (var item in inputsVariables)
721     {
722         item.interactable = false;
723     }
724     //carga el sonido del arreglo numerosGuardados en la posición actual
725     sonidoAS.Stop();
726     //carga el sonido del arreglo numerosGuardados en la posición actual
727     nextFieldProg[numPP].Select(); //selecciona el botón en la posición actual
728     sonidoAS.Play();
729 }
730 // si el nombre del botón en la posición del contenedor numP es igual a 'Salir'
731 if (nextFieldProg[numPP].name == "Salir")
732 {
733     conteoAlter++;
734     //recorro cada elemento de la lista inputsVariables mediante
735     //la variable item y los monto a blancos
736     foreach (var item in inputsVariables)
737     {
738         item.interactable = false;
739     }
740     //carga el sonido del arreglo numerosGuardados en la posición actual
741     sonidoAS.Stop();
742     //carga el sonido del arreglo numerosGuardados en la posición actual
743     nextFieldProg[numPP].Select(); //selecciona el botón en la posición actual
744     sonidoAS.Play();
745 }
746 // si el nombre del botón en la posición del contenedor numP es igual a 'Atras'
747 if (nextFieldProg[numPP].name == "Atras")
748 {
749     conteoAlter++;
750     //recorro cada elemento de la lista inputsVariables mediante
751     //la variable item y los monto a blancos
752     foreach (var item in inputsVariables)
753     {
754         item.interactable = false;
755     }
756     //carga el sonido del arreglo numerosGuardados en la posición actual
757     sonidoAS.Stop();
758     //carga el sonido del arreglo numerosGuardados en la posición actual
759     nextFieldProg[numPP].Select(); //selecciona el botón en la posición actual
760     sonidoAS.Play();
761 }
762 // si el nombre del botón en la posición del contenedor numP es igual a 'titular.txt'
763 if (nextFieldProg[numPP].name == "titular.txt")
764 {
765     conteoAlter++;
766     //recorro cada elemento de la lista inputsVariables mediante
767     //la variable item y los monto a blancos
768     foreach (var item in inputsVariables)
769     {
770         item.interactable = false;
771     }
772     //carga el sonido del arreglo numerosGuardados en la posición 0
773     sonidoAS.Stop();
774     //carga el sonido del arreglo numerosGuardados en la posición 0
775     nextFieldProg[numPP].Select(); //selecciona el botón en la posición 0
776     sonidoAS.Play();
777 }
778 // si el nombre del botón en la posición del contenedor numP es igual a 'titular.txt'
779 if (nextFieldProg[numPP].name == "titular.txt")
780 {
781     conteoAlter++;
782     //recorro cada elemento de la lista inputsVariables mediante
783     //la variable item y los monto a blancos
784     foreach (var item in inputsVariables)
785     {
786         item.interactable = false;
787     }
788     //carga el sonido del arreglo numerosGuardados en la posición 0
789     sonidoAS.Stop();
790     //carga el sonido del arreglo numerosGuardados en la posición 0
791     nextFieldProg[numPP].Select(); //selecciona el botón en la posición 0
792     sonidoAS.Play();
793 }
794 // si el nombre del botón en la posición del contenedor numP es igual a 'desarrolla.txt'
795 if (nextFieldProg[numPP].name == "desarrolla.txt")
796 {
797     conteoAlter++;
798     //recorro cada elemento de la lista inputsVariables mediante
799     //la variable item y los monto a blancos
800     foreach (var item in inputsVariables)
801     {
802         item.interactable = false;
803     }
804     //carga el sonido del arreglo numerosGuardados en la posición 1
805     sonidoAS.Stop();
806     //carga el sonido del arreglo numerosGuardados en la posición 1
807     nextFieldProg[numPP].Select(); //selecciona el botón en la posición 1
808     sonidoAS.Play();
809 }
810 // si el nombre del botón en la posición del contenedor numP es igual a 'desarrolla.txt'
811 if (nextFieldProg[numPP].name == "desarrolla.txt")
812 {
813     conteoAlter++;
814     //recorro cada elemento de la lista inputsVariables mediante
815     //la variable item y los monto a blancos
816     foreach (var item in inputsVariables)
817     {
818         item.interactable = false;
819     }
820     //carga el sonido del arreglo numerosGuardados en la posición 1
821     sonidoAS.Stop();
822     //carga el sonido del arreglo numerosGuardados en la posición 1
823     nextFieldProg[numPP].Select(); //selecciona el botón en la posición 1
824     sonidoAS.Play();
825 }

```

Fig 200"Control por teclado función Tabulando() parte 9"

```

    // si el botón de la posición del contador, numPP es igual al siguiente
    if (nextFieldPreg[numPP].name == "Siguiente")
    {
        audiosConteo[5];
        nextFieldPreg[numPP].Select(); // selecciona el botón en la posición actual.
        sonidoAS.clip = audiosPreg[5]; // carga el audio del arreglo audiosPreg en la posición 5
        sonidoAS.Play();
    }

    // activa la imagen de la ultima posición del arreglo.
    jueg1[numPP - 1].enabled = false;
    // activa la imagen actual.
    jueg1[numPP].enabled = true;

    // si se presiona alguna de estas teclas (flecha izquierda, arriba), se irá moviendo hacia atrás
    else if (Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow) || Input.GetKeyDown(KeyCode.LeftAlt))
    {
        // recorre cada elemento de la lista preguntasSonidos
        // la variable item es el elemento actual
        foreach (var item in preguntasSonidos)
        {
            item.interactable = true;
        }
        // si el contador es menor a 0
        if (conteoAlter < 0)
        {
            conteoAlter = 0; // inicializa el contador en 0
        }
        // si el contador es mayor al tamaño del arreglo
        else if (conteoAlter + preguntasSonidos.Length - 1)
        {
            conteoAlter = 5; // inicializa el contador en 5
        }
        // cuando se detiene las audios de los paneles cuando detecta una entrada por teclado,
        // o de esa manera no se escuchan los audios que estén reproduciéndose
        foreach (var item in audiosPreMouse)
        {
            item.Stop();
        }
        // si el contador numPP es > 0
        if (numPP > 0)
        {
            // si el arreglo guarda en el contador numPreg
            numPP = totPregs;
            // si el arreglo guarda en el contador numPreg
            numPP = totPregs;
            // guarda en el contadorAlter el tamaño del arreglo preguntasSonidos
            conteoAlter = preguntasSonidos.Length - 1;
            // selecciona el botón en la posición actual.
            nextFieldPreg[numPP].Select();
            jueg1[0].enabled = false; // desactiva la imagen de la posición 0.
            jueg1[numPP].enabled = true; // activa la imagen de la posición actual.
            sonidoAS.Stop(); // detiene todos los audios.

            // si el botón siguiente está activado
            if (botSig.activeSelf == true)
            {
                // cuando se guarda el tamaño del arreglo audiosPreg en el contador audiosConteo
                audiosConteo = audiosPreg.Length - 1;
            }
            // si el botón siguiente está desactivado
            else if (botSig.activeSelf == false)
            {
                // cuando se guarda el tamaño del arreglo audiosPreg -2 en el contador audiosConteo
                audiosConteo = audiosPreg.Length - 2;
            }
            sonidoAS.clip = audiosPreg[audiosConteo]; // carga el audio del arreglo audiosPreg en la posición actual de audiosConteo
            sonidoAS.Play(); // comienza a reproducir el audio
        }
    }
}

```

Fig 201 "Control por teclado función Tabulando() parte 10"



```
    int key = System.in.read();
    if(key == '\t') {
        int offset = 4;
        if(tabSize == 2) offset = 2;
        int i = 0;
        while(i < offset) {
            if(cursorIndex + i + 1 > text.length()) break;
            char c = text.charAt(cursorIndex + i + 1);
            if(c == ' ' || c == '\t' || c == '\n') break;
            text = text.substring(0, cursorIndex + i + 1) + " " + text.substring(cursorIndex + i + 1);
            i++;
        }
        cursorIndex += offset;
    }
}

```

Fig 202 "Control por teclado función Tabulando() parte 11"

```

872 //si el nombre del botón en la posición del contador numPP es igual Op3_act
873 if (nextFieldPreg[numPP].name == "Op3_act")
874 {
875     conteoAlter--;//decremento en 1 a la posición del contador.
876
877     /*recorro cada elemento de la lista inputsVarios mediante
878     la variable item y los mando a bloquear*/
879     foreach (var item in inputsVarios){
880         item.interactable = false;
881     }
882     //cargo el audio del arreglo numerosGuardados en la posición 2
883     sonidoAS.clip = preguntasSonidos[numerosGuardados[2]];
884     sonidoAS.Play();
885 }
886 //si el nombre del botón en la posición del contador numPP es igual Op4_act
887 if (nextFieldPreg[numPP].name == "Op4_act")
888 {
889     conteoAlter--;//decremento en 1 a la posición del contador.
890
891     /*recorro cada elemento de la lista inputsVarios mediante
892     la variable item y los mando a bloquear*/
893     foreach (var item in inputsVarios){
894         item.interactable = false;
895     }
896     //cargo el audio del arreglo numerosGuardados en la posición 3
897     sonidoAS.clip = preguntasSonidos[numerosGuardados[3]];
898     sonidoAS.Play();
899 }
900 //si el nombre del botón en la posición del contador numPP es igual Op5_act
901 if (nextFieldPreg[numPP].name == "Op5_act")
902 {
903     conteoAlter--;//decremento en 1 a la posición del contador.
904
905     /*recorro cada elemento de la lista inputsVarios mediante
906     la variable item y los mando a bloquear*/
907     foreach (var item in inputsVarios){
908         item.interactable = false;
909     }
910     //cargo el audio del arreglo numerosGuardados en la posición 4
911     sonidoAS.clip = preguntasSonidos[numerosGuardados[4]];
912     sonidoAS.Play(); //mando a reproducir el audio
913 }
914 //si el nombre del botón en la posición del contador numPP es igual Op6_act
915 if (nextFieldPreg[numPP].name == "Op6_act")
916 {
917     conteoAlter--;//decremento en 1 a la posición del contador.
918
919     /*recorro cada elemento de la lista inputsVarios mediante
920     la variable item y los mando a bloquear*/
921     foreach (var item in inputsVarios){
922         item.interactable = false;
923     }
924     //cargo el audio del arreglo numerosGuardados en la posición 5
925     sonidoAS.clip = preguntasSonidos[numerosGuardados[5]];
926     sonidoAS.Play(); //mando a reproducir el audio
927 }

```

Fig 203"Control por teclado función Tabulando() parte 12"

```

    // si el nombre del botón es la posición del control numPP es igual al numConteo.
    nextFieldPreg[numPP].name == numConteo
    else if (nextFieldPreg[numPP].name == "rop1_inpt" || nextFieldPreg[numPP].name == "rop2_inpt"
    || nextFieldPreg[numPP].name == "rop3_inpt" || nextFieldPreg[numPP].name == "rop4_inpt"
    || nextFieldPreg[numPP].name == "rop5_inpt" || nextFieldPreg[numPP].name == "rop6_inpt")

        // si es que comete la comisión se carga el audio general
        sonidoAS.clip = jeraAud;
        sonidoAS.Play();//cargando el sonido en la memoria

    // si el nombre del botón es la posición del controlador numPP es igual a position_rop.
    position_rop, position_rop1, position_rop2, position_rop3, position_rop4,
    position_rop5, position_rop6
    else if (nextFieldPreg[numPP].name == "rop1dias_inpt" || nextFieldPreg[numPP].name == "rop2dias_inpt"
    || nextFieldPreg[numPP].name == "rop3dias_inpt" || nextFieldPreg[numPP].name == "rop4dias_inpt"
    || nextFieldPreg[numPP].name == "rop5dias_inpt" || nextFieldPreg[numPP].name == "rop6dias_inpt")

        // si es que comete la comisión se carga el audio general
        sonidoAS.clip = diasAud;
        sonidoAS.Play();//cargando el sonido
        nextFieldPreg[numPP].Select();//selecciona el botón en la posición actual.

    // si el control numFPP es igual a position_rop
    if (numFPP == 1).enabled = false; //desactiva la imagen de la posición actual
    jpegI[numFPP].enabled = true; //activa la imagen actual.

    // si el control numFPP es igual a position_rop1
    if (nextFieldPreg[numPP].name == "PreferenciasB.txt")
    {
        audiosConteo[1];//inicializa el contador en 1
        nextFieldPreg[numPP].Select();//selecciona el botón en la posición actual.
        sonidoAS.clip = audiosPreg[1];//carga el audio del arreglo audiosPreg en la posición 1
        sonidoAS.Play();
        jpegI[numFPP + 1].enabled = false; //desactiva la imagen de la posición actual
        jpegI[numFPP].enabled = true; //activa la imagen actual.
    }

    // si el control numFPP es igual a position_rop2
    if (nextFieldPreg[numPP].name == "Salir")
    {
        audiosConteo[0];//inicializa el contador en 0
        nextFieldPreg[numPP].Select();//selecciona el botón en la posición actual.
        sonidoAS.clip = audiosPreg[0];//carga el audio del arreglo audiosPreg en la posición 0
        sonidoAS.Play();
        jpegI[numFPP + 1].enabled = false; //desactiva la imagen de la posición actual
        jpegI[numFPP].enabled = true; //activa la imagen actual.
    }

    // si el control numFPP es igual a position_rop3
    if (nextFieldPreg[numPP].name == "Atras")
    {
        audiosConteo[4];//inicializa el contador en 4
        nextFieldPreg[numPP].Select();//selecciona el botón en la posición actual.
        sonidoAS.clip = audiosPreg[4];//carga el audio del arreglo audiosPreg en la posición 4
        sonidoAS.Play();
        jpegI[numFPP + 1].enabled = false; //desactiva la imagen de la posición actual
        jpegI[numFPP].enabled = true; //activa la imagen actual.
    }
}

```

Fig 204"Control por teclado función Tabulando() parte 13"

```

986     //si el contador numPP es igual totPregs, llega al final del arreglo
987     else if (numPP == totPregs)
988     {
989         /* Si se presiona alguna de estas teclas(tabulador, flechas(derecha, abajo)), irá navegando hacia adelante */
990         if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
991         {
992             //Y si el contador conteoAlter es menor a 0
993             if(conteoAlter < 0){
994
995                 conteoAlter = 0;//initializo el contador en 0
996             }
997             //si el contador es mayor al tamaño del arreglo preguntasSonidos
998             else if(conteoAlter > preguntasSonidos.Length - 1){
999                 conteoAlter = 5;//initializo en contador en 0
999             }
999             /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
999             y de esa manera no se cruzan los audios que estén reproduciéndose*/
999             foreach (var item in audiosPregMouse)
999             {
999                 item.Stop();
999             }
999             juegI[numPP].enabled = false;//desactivo la imagen actual.
999             numPP = 0;//initializo el contador en 0
999             numFPP = 0;//initializo el contador en 0
999             audiosConteo = 0;//initializo el contador en 0
999             conteoAlter = 0;//initializo el contador en 0
999             nextFieldPreg[numPP].Select();//selecciono el botón en la posición actual.
999             sonidoAS.Stop();//detengo todos los audios
999             sonidoAS.clip = audiosPreg[numPP];//cargo el audio del arreglo audiosPreg de la posición actual de numPP,
999             sonidoAS.Play();//mando a reproducir
999             juegI[numPP].enabled = true;//activo la imagen actual.
999         }
999         /* Si se presiona alguna de estas teclas(flechas(izquierda, arriba)), irá navegando hacia atrás */
999         else if(Input.GetKeyDown(KeyCode.LeftArrow)|| Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt))
999         {
999             //si el contador es menor a 0
999             if(conteoAlter < 0){
999                 conteoAlter = 0;//initializo el contador en 0
999             }
999             //si el contador es mayor al tamaño del arreglo preguntasSonidos
999             else if(conteoAlter > preguntasSonidos.Length - 1){
999                 conteoAlter = 5;//initializo el contador en 5
999             }
999             numPP--;//decremento en 1 a la posición del contador.
999             numFPP--;//decremento en 1 a la posición del contador.
999             audiosConteo--;//decremento en 1 a la posición del contador.
999             /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
999             y de esa manera no se cruzan los audios que estén reproduciéndose*/
999             foreach (var item in audiosPregMouse)
999             {
999                 item.Stop();
999             }
999             //desactivo la imagen de la posición actual +1
999             juegI[numPP + 1].enabled = false;
999             nextFieldPreg[numPP].Select();//selecciono el botón en la posición actual.
999             sonidoAS.Stop();//detengo todos los audios

```

Fig 205"Control por teclado función Tabulando() parte 15"

```

1040
1041         nextFieldPreg[numPP].Select();//selecciono el botón en la posición actual.
1042         sonidoAS.Stop();//detengo todos los audios
1043         //si el botón siguiente está activado
1044         if (botSig.activeSelf == true)
1045         {
1046             //cargo el audio del arreglo audiosFin de la posición actual de audiosConteo
1047             sonidoAS.clip = audiosPreg[audiosConteo];
1048
1049             //si el botón siguiente está desactivado
1050             else if (botSig.activeSelf == false)
1051             {
1052                 //cargo el audio diasAud
1053                 sonidoAS.clip = diasAud;
1054             }
1055
1056             sonidoAS.Play();//mando a reproducir el audio
1057             juegI[numFPP].enabled = true;//activo la imagen actual.
1058         }
1059
1060     }
1061
1062 }

```

Fig 206"Control por teclado función Tabulando() parte 16"

# T

```
else if (GeneralLimpiar.menu == "F10A")
{
    if (pregs.activeSelf == false) // si los genericos en las propiedades estan desactivados
        return; // se cancela cuando no se cumple al final del arreglo anteriormente
    if (nextFF < nextFieldFin.length - 1)
    {
        if (!el.contadorDeInputs > 0)
        if (nextFF > 0)
            nextFieldIn[nextFF].Select(); // selecciona el campo en la posicion actual
        finIn[menuFF].enabled = true; // activa la linea de la posicion actual
        if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
            el.subeAumentaLosNivelesDelJuego; // cuando detecta una entrada que cambia
        if (el.subeAumentaLosNivelesDelJuego > 0)
            el.subeAumentaLosNivelesDelJuego -= 1;
        if (el.subeAumentaLosNivelesDelJuego < 0)
            el.subeAumentaLosNivelesDelJuego += 1;
        if (el.subeAumentaLosNivelesDelJuego == 0)
            el.subeAumentaLosNivelesDelJuego = 0;
        if (el.subeAumentaLosNivelesDelJuego > 0)
            sonidoAS.Stop(); // detiene la linea de audio
        sonidoAS.clip = audioSel[nextFF]; // carga el audio del arreglo anterior de la posicion actual del arreglo menu
        sonidoAS.Play(); // inicia o reinicia el audio
        nextFF++; // incrementa el contador en 1
        menuFF++; // incrementa el contador en 1
        nextFieldIn[nextFF].Select(); // selecciona el campo en la posicion actual
        if (nextFF == 0) el.contadorDeInputs = 0;
        if (puntajeFinal == 80) el.sonidoFinal = sonido8;
        sonidoAS.Stop(); // detiene la linea de audio
        sonidoAS.clip = audioSel[menuFF]; // carga el audio del arreglo anterior de la posicion en la posicion actual
        sonidoAS.Play(); // inicia o reinicia el audio
    }
    else if (puntajeFinal == 25) el.sonidoFinal = sonido2;
    sonidoAS.Stop(); // detiene la linea de audio
    sonidoAS.clip = audioSel[menuFF]; // carga el audio del arreglo anterior de la posicion en la posicion actual
    sonidoAS.Play(); // inicia o reinicia el audio
}
else if (puntajeFinal == 50) el.sonidoFinal = sonido5;
sonidoAS.Stop(); // detiene la linea de audio
sonidoAS.clip = audioSel[menuFF]; // carga el audio del arreglo anterior de la posicion en la posicion actual
sonidoAS.Play(); // inicia o reinicia el audio
}
else if (puntajeFinal == 75) el.sonidoFinal = sonido7;
sonidoAS.Stop(); // detiene la linea de audio
sonidoAS.clip = audioSel[menuFF]; // carga el audio del arreglo anterior de la posicion en la posicion actual
sonidoAS.Play(); // inicia o reinicia el audio
}
else if (puntajeFinal == 100) el.sonidoFinal = sonido10;
sonidoAS.Stop(); // detiene la linea de audio
sonidoAS.clip = audioSel[menuFF]; // carga el audio del arreglo anterior de la posicion en la posicion actual
sonidoAS.Play(); // inicia o reinicia el audio
```

Fig 207"Control por teclado función Tabulando() parte 17"

```

1321
1322     else if (numFF < 4) // si el contador de número es 4
1323     {
1324         sonidoAS.Stop(); // detiene todos los sonidos
1325         //carga el audio del arreglo anterior de la posición actual
1326         sonidoAS.clip = audiosFin[numFF];
1327         sonidoAS.Play(); // inicia e reproduce el sonido
1328         finI.numIFF = false; //desactiva la imagen de la última posición del arreglo.
1329         finI.numIFF.enabled = true; //activa la imagen de la posición actual.
1330     }
1331
1332     else if (numFF == 5)
1333     {
1334         sonidoAS.Stop(); // detiene todos los sonidos
1335         sonidoAS.clip = audiosFin[5];
1336         sonidoAS.Play();
1337         //inicializa el contador numFF en 5 en decir seleccionando el último elemento del arreglo,
1338         numFF = nextFieldFin.Length - 1;
1339     }
1340
1341
1342     finI.numIFF = false; //desactiva la imagen de la última posición del arreglo.
1343     finI.numIFF.enabled = true; //activa la imagen actual.
1344
1345     // si se presiona alguno de estos teclas(Flechas Izquierda, flecha arriba, flecha abajo) se habilita este evento
1346     else if (Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow) || Input.GetKeyDown(KeyCode.LeftAlt))
1347     {
1348         //se habilita o deshabilita los sonidos del mouse cuando detecta una entrada por teclado,
1349         //si de esta manera no se activan los sonidos con el ratón se evita contradicciones
1350         foreach (var item in audiosFinMouse)
1351         {
1352             item.Stop();
1353         }
1354
1355         if (numFF == 5)
1356         {
1357             //inicializa el contador numFF en 5 en decir seleccionando el último elemento del arreglo.
1358             numFF = nextFieldFin.Length - 1;
1359             //inicializa el contador numFF en 5 en decir seleccionando el último elemento del arreglo.
1360             numIFF = nextFieldFin.Length - 1;
1361
1362             nextFieldFin[numFF].Select(); //selecciona el botón en la posición actual.
1363             finI[0].enabled = false; //desactiva la imagen en la posición 0.
1364             finI.numIFF.enabled = true; //activa la imagen actual.
1365             conteoAlter = preguntasSonidos.Length - 1;
1366             sonidoAS.Stop(); //detiene todos los sonidos
1367             sonidoAS.clip = audiosFin[audiosFin.Length - 1];
1368             sonidoAS.Play(); //inicia e reproduce el sonido
1369         }
1370         else
1371         {
1372             numFF--; //decrementa en 1 en la posición del contador
1373             numIFF--; //decrementa en 1 en la posición del contador
1374
1375             finI.numIFF > 1 enabled = false; //desactiva la imagen de la posición actual >1
1376             finI.numIFF.enabled = true; //activa la imagen actual.
1377             //si el nombre del botón en la posición del contador numFF es igual a Salir
1378             if (nextFieldFin[numFF].name == "Salir")
1379             {
1380                 sonidoAS.Stop(); //detiene todos los sonidos
1381                 sonidoAS.clip = audiosFin[0]; //carga el audio del arreglo audiosFin posición 0
1382                 sonidoAS.Play(); //inicia e reproduce el sonido
1383             }
1384         }
1385     }

```

Fig 208"Control por teclado función Tabulando() parte 18"

```

1396
1397     }
1398
1399     else if (numFF == nextFieldFin.Length - 1)
1400     {
1401         // si se presiona alguno de estos teclas(Flechas Izquierda, flecha abajo, flecha arriba, flecha derecha) se habilita este evento
1402         // si de esta manera no se activan los sonidos con el ratón se evita contradicciones
1403         foreach (var item in audiosFinMouse)
1404         {
1405             item.Stop();
1406         }
1407
1408         //se habilita o deshabilita los sonidos del mouse cuando detecta una entrada por teclado,
1409         //si de esta manera no se activan los sonidos con el ratón se evita contradicciones
1410         foreach (var item in audiosFinMouse)
1411         {
1412             item.Stop();
1413         }
1414
1415         finI.numIFF.enabled = false;
1416         numFF = 0;
1417         numIFF = 0;
1418         nextFieldFin[numFF].Select(); //selecciona el botón en la posición actual.
1419         sonidoAS.Stop();
1420         sonidoAS.clip = audiosFin[numFF]; //carga el audio del arreglo audiosFin de la posición actual de la posición actual.
1421         sonidoAS.Play();
1422         finI.numIFF.enabled = true; //activa la imagen de la posición actual.
1423     }

```

Fig 209"Control por teclado función Tabulando() parte 19"

```

1219     /* Si se presiona alguna de estas teclas(flechas(izquierda, arriba)), irá navegando hacia atrás */
1220     else if (Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow) || Input.GetKeyDown(KeyCode.LeftAlt))
1221     {
1222         numFF--;
1223         numIFF--;
1224         finI[numIFF + 1].enabled = false; //desactivo la imagen de la posición actual +1.
1225         finI[numIFF].enabled = true; //activo la imagen de la posición actual
1226         nextFieldIn[numFF].Select(); //selecciono el botón en la posición actual.
1227
1228         /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1229         y de esa manera no se cruzan los audios que estén reproduciéndose*/
1230         foreach (var item in audiosFinMouse)
1231         {
1232             item.Stop();
1233         }
1234         //si el puntaje final es 0
1235         if (puntajeFinal == 0)
1236         {
1237             sonidoAS.Stop();
1238             sonidoAS.clip = audiosFin[4]; //cargo el audio del arreglo audiosFin de la posición 4
1239             sonidoAS.Play();
1240         }
1241         //si el puntaje final es 25
1242         else if (puntajeFinal == 25)
1243         {
1244             sonidoAS.Stop();
1245             sonidoAS.clip = audiosFin[5]; //cargo el audio del arreglo audiosFin de la posición 5.
1246             sonidoAS.Play();
1247         }
1248         //si el puntaje final es 50
1249         else if (puntajeFinal == 50)
1250         {
1251             sonidoAS.Stop();
1252             sonidoAS.clip = audiosFin[6]; //cargo el audio del arreglo audiosFin de la posición 6;
1253             sonidoAS.Play();
1254         }
1255         //si el puntaje final es 75
1256         else if (puntajeFinal == 75)
1257         {
1258             sonidoAS.Stop();
1259             sonidoAS.clip = audiosFin[7]; //cargo el audio del arreglo audiosFin de la posición 7.
1260             sonidoAS.Play();
1261         }
1262         //si el puntaje final es 100
1263         else if (puntajeFinal == 100)
1264         {
1265             sonidoAS.Stop();
1266             sonidoAS.clip = audiosFin[8]; //cargo el audio del arreglo audiosFin de la posición 8.
1267             sonidoAS.Play();
1268         }
1269     }
1270 }
1271 }
1272 }
1273 }
```

Fig 210 "Control por teclado función Tabulando() parte 20"

```

1376     else if (GeneralEmpezar.menu == "Fin")
1377     {
1378         //si los gameobjects de las preguntas estan desactivados y el boton siguiente esta activado
1379         if (pregu.activeSelf == false && botSig.activeSelf == true)
1380         {
1381             //si el contador es menor al tamaño del arreglo
1382             if (numFFA < nextFieldFinA.Length - 1)
1383             {
1384                 //si el contador es 0
1385                 if (numFFA == 0)
1386                 {
1387                     //selecciono el boton en la posición actual.
1388                     nextFieldFinA[numFFA].Select();
1389                     //activo la imagen de la posición actual.
1390                     finA[numIFFA].enabled = true;
1391                 }
1392                 /* Si se presiona alguna de estas teclas(tabulador, flechas(derecha, abajo)), irá navegando hacia adelante */
1393                 if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
1394                 {
1395                     /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1396                     y de esa manera no se cruzan los audios que estén reproduciéndose*/
1397                     foreach (var item in audiosFinAMouse)
1398                     {
1399                         item.Stop();
1400                     }
1401                     if (numFFA == 0)
1402                     {
1403                         sonidoAS.Stop();
1404                         //cargo el audio del arreglo audiosFinA de la posición actual del contador numFFA.
1405                         sonidoAS.clip = audiosFinA[numFFA];
1406                         sonidoAS.Play(); //mando a reproducir el audio
1407                     }
1408                     numFFA++; //incremento el contador en 1.
1409                     numIFFA++; //incremento el contador en 1.
1410                     //selecciono el boton en la posición actual.
1411                     nextFieldFinA[numFFA].Select();
1412                     sonidoAS.Stop();
1413                     //cargo el audio del arreglo audiosFinA de la posición actual.
1414                     sonidoAS.clip = audiosFinA[numFFA];
1415                     sonidoAS.Play();
1416                     //desactivo la última imagen del arreglo finA
1417                     finA[numIFFA - 1].enabled = false;
1418                     //activo la imagen de la posición actual.
1419                     finA[numIFFA].enabled = true;
1420                 }
1421                 /* Si se presiona alguna de estas teclas(flechas(síquierda, arriba)), irá navegando hacia atrás */
1422                 else if (Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow) || Input.GetKeyDown(KeyCode.LeftAlt))
1423                 {
1424                     /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1425                     y de esa manera no se cruzan los audios que estén reproduciéndose*/
1426                     foreach (var item in audiosFinAMouse)
1427                     {
1428                         item.Stop();
1429                     }
1430                     if (numFFA == 0)//si el contadore es 0
1431                     {
1432                         //iniciamos el contador numFFA en 5 es decir seleccionamos el ultimo elemento del arreglo.
1433                         numFFA = nextFieldFinA.Length - 1;
1434                         //iniciamos el contadore numIFFA en 5 es decir seleccionamos el ultimo elemento del arreglo.
1435                         numIFFA = nextFieldFinA.Length - 1;
1436                         //selecciono el boton en la posición actual.
1437                         nextFieldFinA[numFFA].Select();
1438                         finA[0].enabled = false;//desactivo la imagen de la posición 0 del arreglo.
1439                         finA[numIFFA].enabled = true;//activo la imagen de la posición actual.
1440                         sonidoAS.Stop();
1441                         //cargo el audio del arreglo audiosFinA de la posición actual.
1442                         sonidoAS.clip = audiosFinA[numFFA];
1443                         sonidoAS.Play();
1444                     }
1445                 }
1446             }
1447         }
1448     }

```

Fig 211"Control por teclado función Tabulando() parte 21"

```

1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
}
else{
    numFFA--;
    numIFFA--;
    //selecciono el boton en la posición actual.
    nextFieldFinA[numFFA].Select();
    sonidoAS.Stop();
    sonidoAS.clip = audiosFinA[numFFA];
    sonidoAS.Play();
    //desactivo la imagen de la posición actual +1;
    finAI[numIFFA + 1].enabled = false;
    //activó la imagen de la posición actual.
    finAI[numIFFA].enabled = true;
}

}
else if (numFFA == nextFieldFinA.Length - 1)//si el controlador es igual al tamaño del arreglo
{
    /* Si se presiona alguna de estas teclas(tabulador, flechas(derecha, abajo)), irá navegando hacia adelante */
    if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
    {
        /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
        y de esa manera no se cruzan los audios que estén reproduciéndose*/
        foreach (var item in audiosFinAMouse)
        {
            item.Stop();
        }
        finAI[numIFFA].enabled = false;//desactivo la imagen de la posición actual.
        numFFA = 0;
        numIFFA = 0;
        nextFieldFinA[numFFA].Select(); //selecciona el botón en la posición actual.
        sonidoAS.Stop();
        sonidoAS.clip = audiosFinA[numFFA]; //cargo el audio del arreglo audiosFinA de la posición actual.
        sonidoAS.Play();
        finAI[numIFFA].enabled = true; //activó la imagen de la posición actual +1.
    }
    /*Si presiona las siguientes teclas
    else if(Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt)){*/
        /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
        y de esa manera no se cruzan los audios que estén reproduciéndose*/
        foreach (var item in audiosFinAMouse)
        {
            item.Stop();
        }
        numFFA--;
        numIFFA--;
        nextFieldFinA[numFFA].Select(); //selecciona el botón en la posición actual.
        sonidoAS.Stop();
        sonidoAS.clip = audiosFinA[numFFA]; //cargo el audio del arreglo audiosFinA de la posición actual.
        sonidoAS.Play();
        finAI[numIFFA+1].enabled = false;//desactivo la imagen de la posición actual +1.
        finAI[numIFFA].enabled = true; //activó la imagen de la posición actual +1.
    }
}
}

```

Fig 212"Control por teclado función Tabulando() parte 22"

```

1481     }
1482     if (GeneralEmpezar.menu == "Preferencias")
1483     {
1484         if (prefer.activeSelf == true)//si el panel de preferencias esta activo,
1485         {
1486             //si el contador numCP es menor al tamaño del arreglo:
1487             if (numCP < nextFieldPrefer.Length - 1)
1488             {
1489                 if (numCP == 0)//si el contador es 0
1490                 {
1491                     nextFieldPrefer[numCP].Select();//selecciona el boton en la posición actual,
1492                     prefImg[numFP].enabled = true;//activa la imagen de la posición actual,
1493                 }
1494             /* Si se presiona alguna de estas teclas tabulador, flechas(derecha, abajo), irá navegando hacia adelante */
1495             if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
1496             {
1497                 /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1498                 y de esa manera no se cruzan los audios que estén reproduciéndose*/
1499                 foreach (var item in audiosPreferMouse)
1500                 {
1501                     item.Stop();
1502                 }
1503                 if (numCP == 0)
1504                 {
1505                     sonidoAS.Stop();//detengo todos los audios,
1506                     sonidoAS.clip = audiosPrefer[numCP];//cargo el audio del arreglo audiosFinal de la posición actual,
1507                     sonidoAS.Play();//mando a reproducir el audio.
1508                 }
1509                 numCP++;
1510                 numFP++;
1511                 sonidoAS.Stop();//detengo todos los audios,
1512                 sonidoAS.clip = audiosPrefer[numCP];//cargo el audio del arreglo audiosFinal de la posición actual,
1513                 sonidoAS.Play();//mando a reproducir el audio.
1514                 nextFieldPrefer[numCP].Select();//selecciona el boton en la posición actual,
1515                 prefImg[numFP - 1].enabled = false;//desactivo la ultima imagen del arreglo,
1516                 prefImg[numFP].enabled = true;//activo la imagen de la posición actual.
1517             }
1518             /*Si presiona las siguientes teclas
1519             else if(Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow)||| Input.GetKeyDown(KeyCode.LeftAlt)){*/
1520                 /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1521                 y de esa manera no se cruzan los audios que estén reproduciéndose*/
1522                 foreach (var item in audiosPreferMouse)
1523                 {
1524                     item.Stop();
1525                 }
1526                 if (numCP == 0)
1527                 {
1528                     //iniciamos el contador numCP en 11 es decir seleccionamos el ultimo elemento del arreglo.
1529                     numCP = nextFieldPrefer.Length - 1;
1530                     //iniciamos el contador numCP en 11 es decir seleccionamos el ultimo elemento del arreglo.
1531                     numFP = nextFieldPrefer.Length - 1;
1532                     nextFieldPrefer[numCP].Select();//selecciona el boton en la posición actual.
1533                     sonidoAS.Stop();//detengo todos los audios,
1534                     sonidoAS.clip = audiosPrefer[numCP];//cargo el audio del arreglo audiosFinal de la posición actual,
1535                     sonidoAS.Play();//mando a reproducir el audio.
1536                     prefImg[0].enabled = false;//desactivo la imagen de la posición 0 del arreglo,
1537                     prefImg[numFP].enabled = true;//activo la imagen de la posición actual.
1538                 }
1539             else if (numCP < nextFieldPrefer.Length - 1)//si el contador es menor al tamaño del arreglo
1540             {
1541                 numCP--;//Hago un decremento en i el contador,
1542                 numFP--;//hago un decremento en i el contador,
1543                 nextFieldPrefer[numCP].Select();//selecciona el boton en la posición actual,
1544                 sonidoAS.Stop();//detengo todos los audios,
1545                 sonidoAS.clip = audiosPrefer[numCP];
1546                 sonidoAS.Play();//mando a reproducir el audio.
1547                 prefImg[numFP + 1].enabled = false;//desactivo la imagen de la posición actual +1,
1548                 prefImg[numFP].enabled = true;//activo la imagen de la posición actual.
1549             }
1550         }
1551     }

```

Fig 213"Control por teclado función Tabulando() parte 23"

```

1473     else if (numCP == nextFieldPrefer.Length - 1)
1474     {
1475         /* Si se presiona alguna de estas teclas (tabulador, flechas/derecha, abajo), irá navegando hacia adelante */
1476         if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
1477         {
1478             /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1479             y de esa manera no se cruzan los audios que estén reproduciéndose */
1480             foreach (var item in audiosPreferMouse)
1481             {
1482                 item.Stop();
1483             }
1484             prefImg[numFP].enabled = false; //desactivo la imagen de la posición actual.
1485             numCP = 0;
1486             numFP = 0;
1487             sonidoAS.Stop(); //detengo todos los audios.
1488             nextFieldPrefer[numCP].Select(); //selecciono el botón en la posición actual.
1489             prefImg[numFP].enabled = true; //activo la imagen de la posición actual.
1490             sonidoAS.clip = audiosPrefer[numCP]; //cargo el audio del arreglo audiosFinA de la posición actual.
1491             sonidoAS.Play();
1492         }
1493         /*Si presiona las siguientes teclas:
1494         else if(Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt)){*/
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504         numCP--;
1505         numFP--;
1506         /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1507         y de esa manera no se cruzan los audios que estén reproduciéndose */
1508         foreach (var item in audiosPreferMouse)
1509         {
1510             item.Stop();
1511         }
1512         prefImg[numFP+1].enabled = false; //desactivo la imagen de la posición actual.
1513         nextFieldPrefer[numCP].Select(); //selecciono el botón en la posición actual.
1514         sonidoAS.Stop(); //detengo todos los audios.
1515         sonidoAS.clip = audiosPrefer[numCP]; //cargo el audio del arreglo audiosFinA de la posición actual.
1516         sonidoAS.Play();
1517         prefImg[numFP].enabled = true; //activo la imagen de la posición actual.
1518     }
1519 }
1520 }

```

Fig 214"Control por teclado función Tabulando() parte 23"

```

1517     /*Esta función me permite poner el cronómetro para poder inicializar
1518     cada vez que respondo las preguntas*/
1519     public void correTiempo()
1520     {
1521         float TimerControl = Time.time - StartTime;
1522         string mins = ((int)TimerControl / 60).ToString("00");
1523         string segs = (TimerControl % 60).ToString("00");
1524         string milisegs = ((TimerControl * 100) % 100).ToString("00");
1525         string TimerString = string.Format("{00}:{01}:{02}", mins, segs, milisegs);
1526         tiempoText.text = TimerString; //en esta Variable se guarda el tiempo de respuesta empleado por pregunta.
1527     }

```

Fig 215"Función correTiempo()"

```

1529 //Esta función es llamada cuando se presiona el botón empezar
1530 public void Empiezo()
1531 {
1532     nuevoJuego(); //inicializa todo como un juego nuevo.
1533     textoTit.text = "EJERCITARIO EL TIEMPO"; //se muestra en el panel main
1534     textodetall.text = "INSTRUCCIONES PARA EL PARTICIPANTE"; //se muestra en el panel main
1535     textocnten.text = instrucciones[conteoInstruc]; //
1536     datetime = DateTime.Now.ToString("hh:mm:ss"); //registrar la hora de inicio
1537     //la imagen de la casa la cual corresponde al botón de reiniciar cambia por la imagen del botón siguiente.
1538     sigoFin.image.sprite = ImagenSiguiente;
1539     GenerarAleatoriosSinRepetir(); //llamo al metodo de generar números aleatorios.
1540     GeneralEmpezar.muevoCamara = false; //deshabilito el movimiento de la cámara.
1541     empiezo.enabled = false; //desactivo el botón empezar.
1542     textoEmpiezo.gameObject.SetActive(false); //desactivo el texto del botón empezar
1543     /*Activo todos los objetos del panel de instrucciones*/
1544     juego.SetActive(true);
1545     botSig.SetActive(true);
1546     botAtras.SetActive(true);
1547     botPref.SetActive(true);
1548     botSalir.SetActive(true);
1549     tit.SetActive(true);
1550     detall.SetActive(true);
1551     conten.SetActive(true);
1552     GeneralEmpezar.menu = "Jugando"; //menu es igual a Jugando
1553 }

```

Fig 216"Función Empiezo()"

```

1555 /*Funcion para generar numeros aleatorios*/
1556 void GenerarAleatoriosSinRepetir()
1557 {
1558     numerosGuardados = new List<int>(); //crea una lista de numeros aleatorios guardados
1559     numPares = new List<int>(); //se crea una lista de numeros pares correspondiente a las preguntas
1560     listaFinalPregs = new List<int>(); //se crea lista de preguntas randomicas.
1561     int posicionAleatoria;
1562     //si el contador es menor al tamaño de la lista contenPregFinal es decir al número de preguntas
1563     for (int i = 0; i < (contenPregFinal.Count); i++)
1564     {
1565         do
1566         {
1567             /* mando a generar numeros aleatorios de 0 a 5, ya que contenPregFinal tiene 6 preguntas*/
1568             posicionAleatoria = Random.Range(0, (contenPregFinal.Count));
1569             //mientras numerosGuardados contenga un numero de posicionAleatoria
1570             } while (numerosGuardados.Contains(posicionAleatoria));
1571             numerosGuardados.Add(posicionAleatoria); //se agrega la posicionAleatoria al arreglo numerosGuardados
1572     }
1573     //recorra con un bucle y digo si i es menor al número de numerosGuardados y le incremente en 1
1574     for (int i = 0; i < numerosGuardados.Count; i++)
1575     {
1576         //añade los numeros guardados al arreglo contenPreg en el orden que recuperé el contador i
1577         contenPreg.Add(contenPregFinal[numerosGuardados[i]]);
1578         //añade los numeros guardados al arreglo alternaJera en el orden que recuperé el contador i
1579         alternaJera.Add(alternaJeraFinal[numerosGuardados[i]]);
1580         //añade los numeros guardados al arreglo alternaDias en el orden que recuperé el contador i
1581         alternaDias.Add(alternaDiasFinal[numerosGuardados[i]]);
1582     }
1583 }

```

Fig 217"Función GenerarAleatoriosSinRepetir()"

```

1586 public void ActivoPrefer()
1587 {
1588     if (prefMuestro == false)//si prefMuestro es false
1589     {
1590         prefer.SetActive(true);//mando a activar el panel de preferencias
1591         GeneralEmpezar.menu = "Preferencias";//menu es igual a Jugando
1592         tit.SetActive(false);//desactivo el titulo para evitar que se choquen con el panel de preferencias.
1593         detail.SetActive(false); //desactivo el detalle para evitar que se choquen con el panel de preferencias.
1594
1595         GeneralEmpezar.prefMuestro = true;//se activa el botón de preferencias.
1596         if (pregu.activeSelf == true)//si mis preguntas estan activadas
1597         {
1598             /* mando a desactivar todas las imagenes del panel de preguntas*/
1599             foreach (var item in juegI)
1600             {
1601                 item.enabled = false;
1602             }
1603         }
1604         //Si el texto de la pregunta es igual a
1605         else if (textodetail.text == "INSTRUCCIONES PARA EL PARTICIPANTE")
1606         {
1607             /* mando a desactivar todas las imagenes del panel de instrucciones*/
1608             foreach (var item in dial)
1609             {
1610                 item.enabled = false;
1611             }
1612         }
1613         //Si el texto de la pregunta es igual a
1614         else if (textodetail.text == "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL")
1615         {
1616             /* mando a desactivar todas las imagenes del panel FinA de calificación*/
1617             foreach (var item in finAI)
1618             {
1619                 item.enabled = false;
1620             }
1621         }
1622         //Si el texto de la pregunta es igual a
1623         else if (textodetail.text == "RETROALIMENTACIÓN FINAL")
1624         {
1625             /* mando a desactivar todas las imagenes del panel Fin*/
1626             foreach (var item in finI)
1627             {
1628                 item.enabled = false;
1629             }
1630         }
1631     }
1632 }
1633 }
1634

```

Fig 218"Función ActivoPrefer()"

```
1635 public void DesctivoPrefer()
1636 {
1637     if (prefMuestro == true)//si el panel de preferencias está activo
1638     {
1639         prefer.SetActive(false);//desactiva el panel de preferencias
1640         tit.SetActive(true);//activa el título
1641         detall.SetActive(true);//activa el detalle
1642         GeneralEmpezar.prefMuestro = false;//desactiva botón de
1643         if (pregu.activeSelf == true)//si las preguntas están activadas
1644         {
1645             //el menu es igual a Preguntas
1646             GeneralEmpezar.menu = "Preguntas";
1647         }
1648         //Si el texto de la pregunta es igual a
1649         else if (textodetail.text == "INSTRUCCIONES PARA EL PARTICIPANTE")
1650         {
1651             //el menu es igual a Jugando
1652             GeneralEmpezar.menu = "Jugando";
1653         }
1654         //Si el texto de la pregunta es igual a
1655         else if (textodetail.text == "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL")
1656         {
1657             //el menu es igual a FinA
1658             GeneralEmpezar.menu = "FinA";
1659         }
1660         //Si el texto de la pregunta es igual a
1661         else if (textodetail.text == "RETROALIMENTACIÓN FINAL")
1662         {
1663             //el menu es igual a Fin
1664             GeneralEmpezar.menu = "Fin";
1665         }
1666     }
1667 }
```

Fig 219"Función DesctivoPrefer()"

```

1695     public void siguiente()
1696     {
1697         ReinicioSeleccion(); //llamo a la función para que cada vez que presione el botón siguiente reinicie los valores a 0.
1698         /* Solo si el panel de preferencias está desactivado podra pasar al siguiente panel al presionar la tecla siguiente . caso contrario deberá cerrar el panel de preferencias primero*/
1699         if (prefer.activeSelf == false) //si el panel de preferencias está activo
1700         {
1701             conteoInstruc++; //incremento el contador en 1
1702             //si el contador es menor al número de instrucciones
1703             if (conteoInstruc < instrucciones.Count)
1704             {
1705                 //se pasa a textoconten la instrucciones de la posición actual del contador conteoInstruc
1706                 textoConten.text = instrucciones[conteoInstruc];
1707             }
1708             //si menú es igual a Fin
1709             else if (GeneralEmpezar.menu == "Fin")
1710             {
1711                 GeneralEmpezar.muevoCamara = true; //activo el movimiento de la cámara con el mouse.
1712                 empiezo.enabled = true; //activo el botón empezar.
1713                 textoEmpiezo.gameObject.SetActive(true); //activo el texto del botón empiezo
1714                 tiempoText.text = "00:00:00"; //inicializo la variable de cronómetro a 0
1715                 juego.SetActive(false);
1716                 botSig.SetActive(false);
1717                 botAtras.SetActive(false);
1718                 botPref.SetActive(false);
1719                 botSalir.SetActive(false);
1720                 tit.SetActive(false);
1721                 detall.SetActive(false);
1722                 conten.SetActive(false);
1723                 prefer.SetActive(false);
1724                 foreach (var item in finI)
1725                 {
1726                     item.enabled = false;
1727                 }
1728                 GeneralEmpezar.menu = "Main";
1729                 pos = 0;
1730                 contRes = 0;
1731                 numC = 0;
1732                 numI = 0;
1733                 numCP = 0;
1734                 numFP = 0;
1735                 numPP = 0;
1736                 numFPP = 0;
1737                 numFF = 0;
1738                 numIFF = 0;
1739                 puntajeFinal = 0;
1740                 conteoAlter = 0;
1741                 audiosConteo = 0;
1742             }
1743             else
1744             {
1745                 conten.SetActive(false);
1746                 nextFieldDiag[0].Select();
1747                 botSig.SetActive(false);
1748                 botAtras.SetActive(true);
1749                 foreach (var item in juegI)
1750                 {
1751                     item.enabled = false;
1752                 }
1753             }
1754         }
1755     }

```

Fig 220"Función Siguiente() parte 1"

```

1730
1731 <
1732     if (pos < contenPreg.Count)
1733     {
1734         GeneralEmpezar.menu = "Preguntas";
1735         //textodetall.fontSize = 12;
1736         textodetall.text = "Priorizar del 1 al 6, considerando que 1, es la actividad más importante, y 6 la menos importante";
1737         pregue.SetActive(true);
1738         textoOp1.text = contenPreg[pos];
1739         textoOp2.text = contenPreg[pos + 1];
1740         textoOp3.text = contenPreg[pos + 2];
1741         textoOp4.text = contenPreg[pos + 3];
1742         textoOp5.text = contenPreg[pos + 4];
1743         textoOp6.text = contenPreg[pos + 5];
1744         pos = pos + 6;
1745     }
1746     else if (pos == contenPreg.Count)
1747     {
1748         GeneralEmpezar.menu = "FinA";
1749         //textofinal = "FinA";
1750         textoSig.text = "Terminar";
1751         //textodetall.fontSize = 24;
1752         textodetall.text = "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL";
1753         if (puntajeFinal == 0)
1754         {
1755             puntajeFinal = 0;
1756         }
1757         else if (puntajeFinal >= 1 && puntajeFinal <= 5)
1758         {
1759             puntajeFinal = 25;
1760             retFin = "Jerarquiza y asigna tiempo a las actividades solo a pocas actividades.\n\nUsted ha obtenido pocos aciertos.\n\n";
1761         }
1762         else if (puntajeFinal == 6)
1763         {
1764             puntajeFinal = 50;
1765             retFin = "Jerarquiza y asigna tiempo a las actividades solo la mitad de las actividades.\n\nUsted ha obtenido la mitad de los aciertos.\n\n";
1766         }
1767         else if (puntajeFinal >= 7 && puntajeFinal <= 11)
1768         {
1769             puntajeFinal = 75;
1770             retFin = "Jerarquiza y asigna tiempo a las actividades. Usted ha obtenido casi todos los aciertos, por favor revise sus estrategias.\n\n";
1771         }
1772         else if (puntajeFinal == 12)
1773         {
1774             puntajeFinal = 100;
1775             retFin = "Jerarquiza y asigna tiempo a las actividades. Usted ha obtenido todos los aciertos. Felicitaciones.";
1776         }
1777         textoconten.text = "Usted ha obtenido el " + (puntajeFinal) + "% en este simulador laboral\n" + retFin;
1778         conten.SetActive(true);
1779         detall.SetActive(true);
1780         pregue.SetActive(false);
1781         botAtras.SetActive(false);
1782         botSig.SetActive(true);
1783         pos++;
1784     }

```

Fig 221"Función Siguiente() parte 2"

```

1001 if (GeneralEmpezar.menu == "Fin" && pos < contenPreg.Count)
1002 {
1003     GeneralEmpezar.menu = "Fin";
1004     //textofinal = "Fin";
1005     signIn.Image.sprite = ImagesCase;
1006     textoSig.text = "Terminar";
1007
1008     textodetall.text = "REBALIZACIÓN FINAL";
1009     textoconten.text = "Capacidad de planificación y manejo del tiempo es: \nDeterminar eficientemente metas y especificar las etapas, acciones, plazos y recursos requeridos";
1010     string ejercicio = DateTime.Now.ToString("MM");
1011     string ejercicio2 = DateTime.Now.Year.ToString("yy");
1012     string ejercicio3 = DateTime.Now.Month.ToString("00") + "-" + System.DateTime.Now.Month.ToString("00") + "-" + System.DateTime.Now.Day.ToString("00") + "-" + System.DateTime.Now.Hour.ToString("00") + ":" + System.DateTime.Now.Minute.ToString("00") + ":" + System.DateTime.Now.Second.ToString("00"));
1013
1014     foreach (KeyValuePair<string, string> item in respuestasDadas)
1015     {
1016         List<string> contex = new List<string>();
1017         item.Value.Split(new string[] { "||" }, StringSplitOptions.None).ToList();
1018         pregue = new Preguntas();
1019         pregue.setContenido(item.Key);
1020         pregue.setTiempoRespuesta(contex[0]);
1021         pregue.setAcuerdoPregunta(Int32.Parse(contex[1]));
1022         ejercicio.setPreguntas(pregue);
1023
1024         conten.SetActive(true);
1025         detall.SetActive(true);
1026         pregue.SetActive(false);
1027         botAtras.SetActive(false);
1028         botSig.SetActive(true);
1029
1030         ejercicio.setTiempo(ejercicio2);
1031         ejercicio.setNumeroDePreguntas(mData.CantidadPreguntas());
1032         ejercicio.setCorreosWebData(mail);
1033
1034         ejercicio.setTiempo((System.DateTime.Now.Hour.ToString("00") + ":" + System.DateTime.Now.Minute.ToString("00")) + ":" + System.DateTime.Now.Second.ToString("00"));
1035         ejercicio.setFechaActual((System.DateTime.Now.Year.ToString("yy")) + "-" + System.DateTime.Now.Month.ToString("00") + "-" + System.DateTime.Now.Day.ToString("00") + "-" + System.DateTime.Now.Hour.ToString("00") + ":" + System.DateTime.Now.Minute.ToString("00") + ":" + System.DateTime.Now.Second.ToString("00"));
1036     }
1037 }

```

Fig 222"Función Siguiente() parte 3"

```

1839     public void Atras()
1840     {
1841         ReinicioSeleccion(); //mando a reiniciar los valores de las preguntas
1842         if (prefer.activeSelf == false) //si es que el panel de preferencias esta desactivado
1843         {
1844             sonidoAS.Stop(); //detengo todos los audios
1845             contRes--; //décemento en 1 a la posición del contador.
1846             pos = 0;
1847             if (pos == 0) //si la posición de la pregunta es 0
1848             {
1849                 /*limpio todas las respuestas ingresadas*/
1850                 resOp1.text = "";
1851                 resOp2.text = "";
1852                 resOp3.text = "";
1853                 resOp4.text = "";
1854                 resOp5.text = "";
1855                 resOp6.text = "";
1856                 diasOp1.text = "";
1857                 diasOp2.text = "";
1858                 diasOp3.text = "";
1859                 diasOp4.text = "";
1860                 diasOp5.text = "";
1861                 diasOp6.text = "";
1862                 puntajeFinal = 0;
1863                 //active
1864                 juego.SetActive(true);
1865                 botSig.SetActive(true);
1866                 botPref.SetActive(true);
1867                 botSalir.SetActive(true);
1868                 botAtras.SetActive(true);
1869                 tit.SetActive(true);
1870                 detall.SetActive(true);
1871                 conten.SetActive(true);
1872                 tiempoText.text = "00:00:00"; //inicio el cronometro en 0
1873                 //estoy en el panel de INSTRUCCIONES
1874                 GeneralEmpezar.menu = "Jugando";
1875                 textoDetail.text = "INSTRUCCIONES PARA EL PARTICIPANTE";
1876                 pregu.SetActive(false); //desactivo las preguntas
1877
1878                 //recorro cada elemento de la lista inputsVarios mediante
1879                 //la variable item y les mando a activar //
1880                 foreach (var item in inputsVarios)
1881                 {
1882                     item.interactable = true;
1883                 }
1884                 //décemento en 1 a la posición del contador.
1885                 conteoInstruc--;
1886                 //si el contador es Mayor a -1
1887                 if (conteoInstruc > -1)
1888                 {
1889                     //cargo los instrucciones
1890                     textoConten.text = instrucciones[conteoInstruc];
1891                 }
1892                 else
1893                 {
1894                     //Desactivo todo
1895                     juego.SetActive(false);
1896                     botSig.SetActive(false);
1897                     botPref.SetActive(false);
1898                     botSalir.SetActive(false);
1899                     botAtras.SetActive(false);
1900                     tit.SetActive(false);
1901                     detall.SetActive(false);
1902                     conten.SetActive(false);
1903                     //estoy en el panel principal
1904                     GeneralEmpezar.menu = "Main";
1905                     //El movimiento de la cámara se activa
1906                     GeneralEmpezar.muevoCamara = true;
1907                     //esta activo el botón empieza
1908                     empiezo.enabled = true;
1909                     textoEmpiezo.gameObject.SetActive(true);
1910                 }
1911             }
1912         }
1913     }

```

Fig 223 "Función Atras()"

Esta función permite validar que los valores ingresados por teclado sean números y si ingresa otros valores se manda a reproducir un audio en el que le indica al participante que solo debe ingresar números.

```

1912     /*función para validar que se ingresen solo números como respuestas*/
1913     public void soloNumeros(TMP_InputField textBox1)
1914     {
1915         try
1916         {
1917             //convierbo a texto el valor ingresado
1918             int.Parse(textBox1.text);
1919         }
1920         catch (Exception e)
1921         {
1922             //si ingresa un valor no numerico es limpiará el objeto
1923             textBox1.text = "";
1924         }
1925     }

```

Fig 224 "Función soloNumeros()"

Esta función permite controlar que el no se ingresen números repetidos por teclado, esto se hace ya que son valores para jerarquizar prioridad de actividades y cuando se trate de ingresar un valor ya repetido se reproducirá un audio indicando que el número ingresado es repetido.

```

1926     /*Función para validar que no se ingresen números repetidos*/
1927     public void noRepetidosIngreso(TMP_InputField textBox1)
1928     {
1929         /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1930         y de esa manera no se cruzan los audios que estén reproduciéndose*/
1931         foreach (var item in audiosPregMouse)
1932         {
1933             item.Stop();
1934         }
1935         /*mando a recorrer cada valor del arreglo y digo
1936         que si mi item es diferente del valor que se está ingresando y si el valor de item
1937         es igual al valor ingresado y es no está vacío mando a reproducir
1938         el audio de que el valor ingresado es repetido */
1939         foreach (var item in jerarquiaInput)
1940         {
1941             if (item.name != textBox1.name)
1942             {
1943                 if (item.text == textBox1.text && textBox1.text != "")
1944                 {
1945                     textBox1.text = "";//limpio el valor ingresado
1946                     sonidoAS.clip = avisoRepe;//cargo el audio de aviso de número repetido
1947                     sonidoAS.Play();//mando a reproducir el audio
1948                 }
1949             }
1950         }
1951     }

```

Fig 225 "Función noRepetidosIngreso()"

Esta función controla que para las respuestas de jerarquizar se ingresen solo valores de 1 a 6, caso contrario, se limpia el valor ingresado y a la vez se reproduce un audio en el que le indica que solo debe ingresar valores de 1 a 6,

```

1953     /*Función para validar que solo se ingresen valores no mayores a 6*/
1954     public void soloHastaSeis(TMP_InputField textBox1)
1955     {
1956         /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1957         y de esa manera no se cruzan los audios que estén reproduciéndose*/
1958         foreach (var item in audiosPregMouse)
1959         {
1960             item.Stop();
1961         }
1962         try
1963         {
1964             //convierto el string a entero
1965             int.Parse(textBox1.text);
1966             //si el valor ingresado es mayor a 6
1967             if (int.Parse(textBox1.text) > 6)
1968             {
1969                 //limpio el campo de texto
1970                 textBox1.text = "";
1971                 //cargo el audio de aviso de que el número es mayor a 6
1972                 sonidoAS.clip = avisoMayor;
1973                 sonidoAS.Play(); //mando a reproducir el audio
1974             }
1975         }
1976         catch (Exception e)
1977         {
1978             //limpio el campo de texto
1979             textBox1.text = "";
1980         }
1981     }

```

Fig 226 "Función soloHastaSeis()"



Fig 227 "Las tres últimas funciones son llamadas dentro de cada objeto de respuesta"

```

981 public void Respondo(TMP_InputField textoBot)
982 {
983     //si el nombre del botón es igual a rop1_inpt
984     if (textoBot.name == "rop1_inpt")
985     {
986         /*Compró si el valor ingresado es igual a la respuesta que se tiene en
987         la posición 0, si es que son igual es puntaje sube en 1*/
988         if (textoBot.text == alternaJera[0])
989         {
990             puntajeFinal++;
991         }
992         /*Y en la posición 0 de mi arreglo de respuestasDadas agrego el valor ingresado + el tiempo de respuesta
993         + el número de pregunta que se encuentre en la posición 0 del arreglo numerosGuardados+1*/
994         respuestasDadas[0] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[0]+1);
995     }
996     //si el nombre del botón es igual a rop1dias_inpt
997     else if (textoBot.name == "rop1dias_inpt")
998     {
999         /*Compró si el valor ingresado es igual a la respuesta que se tiene en
1000         la posición 0, si es que son iguales el puntaje sube en 1*/
1001         if (textoBot.text == alternaDias[0])
1002         {
1003             puntajeFinal++;
1004         }
1005         /*Y en la posición 1 de mi arreglo de respuestasDadas agrego el valor ingresado + el tiempo de respuesta
1006         + la pregunta que se encuentre en la posición 0 del arreglo numerosGuardados+7*/
1007         respuestasDadas[1] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[0]+7);
1008     }
1009     //si el nombre del botón es igual a rop2_inpt
1010     else if (textoBot.name == "rop2_inpt")
1011     {
1012         /*Compró si el valor ingresado es igual a la respuesta que se tiene en
1013         la posición 1, si es que son igual es puntaje sube en 1 */
1014         if (textoBot.text == alternaJera[1])
1015         {
1016             puntajeFinal++;
1017         }
1018         /*Y en la posición 2 de mi arreglo de respuestasDadas agrego el valor ingresado + el tiempo de respuesta
1019         + la pregunta que se encuentre en la posición 1 del arreglo numerosGuardados+1*/
1020         respuestasDadas[2] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[1]+1);
1021     }
1022     //si el nombre del botón es igual a rop2dias_inpt
1023     else if (textoBot.name == "rop2dias_inpt")
1024     {
1025         /*Compró si el valor ingresado es igual a la respuesta que se tiene en
1026         la posición 1, si es que son iguales el puntaje sube en 1*/
1027         if (textoBot.text == alternaDias[1])
1028         {
1029             puntajeFinal++;
1030         }
1031         /*Y en la posición 3 de mi arreglo de respuestasDadas agrego el valor ingresado + el tiempo de respuesta
1032         + la pregunta que se encuentre en la posición 1 del arreglo numerosGuardados+7*/
1033         respuestasDadas[3] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[1]+7);
1034     }
1035 }
1036 }
```

*Fig 228"Fragmento de código de la función Respondo() parte 1"*

```

2037 //si el nombre del botón es igual a rop3_inpt
2038 else if (textoBot.name == "rop3_inpt")
2039 {
2040     /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
2041     la posición 2, si es que son iguales el puntaje sube en 1*/
2042     if (textoBot.text == alternaJera[2])
2043     {
2044         puntajeFinal++;
2045     }
2046     /*Y en la posición 4 de mi arreglo de respuestasDadas agrego el valor ingresado + el tiempo de respuesta
2047     + la pregunta que se encuentre en la posición 2 del arreglo numerosGuardados+1*/
2048     respuestasDadas[4] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[2] + 1);
2049 }
2050 //si el nombre del botón es igual a rop3dias_inpt
2051 else if (textoBot.name == "rop3dias_inpt")
2052 {
2053     /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
2054     la posición 2, si es que son iguales el puntaje sube en 1*/
2055     if (textoBot.text == alternaDias[2])
2056     {
2057         puntajeFinal++;
2058     }
2059     /*Y en la posición 5 de mi arreglo de respuestasDadas agrego el valor ingresado + el tiempo de respuesta
2060     + la pregunta que se encuentre en la posición 2 del arreglo numerosGuardados+7*/
2061     respuestasDadas[5] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[2] + 7);
2062 }
2063 //si el nombre del botón es igual a rop4_inpt
2064 if (textoBot.name == "rop4_inpt")
2065 {
2066     /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
2067     la posición 3, si es que son iguales el puntaje sube en 1*/
2068     if (textoBot.text == alternaJera[3])
2069     {
2070         puntajeFinal++;
2071     }
2072     /*Y en la posición 6 de mi arreglo de respuestasDadas agrego el valor ingresado + el tiempo de respuesta
2073     + la pregunta que se encuentre en la posición 3 del arreglo numerosGuardados+1*/
2074     respuestasDadas[6] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[3] + 1);
2075 }
2076 //si el nombre del botón es igual a rop4dias_inpt
2077 else if (textoBot.name == "rop4dias_inpt")
2078 {
2079     /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
2080     la posición 3, si es que son iguales el puntaje sube en 1*/
2081     if (textoBot.text == alternaDias[3])
2082     {
2083         puntajeFinal++;
2084     }
2085     /*Y en la posición 7 de mi arreglo de respuestasDadas agrego el valor ingresado + el tiempo de respuesta
2086     + la pregunta que se encuentre en la posición 3 del arreglo numerosGuardados+7*/
2087     respuestasDadas[7] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[3] + 7);
2088 }

```

Fig 229"Fragmento de código de la función Respondo() parte 2"

```

2089 //si el nombre del botón es igual a rop5_inpt
2090 if (textoBot.name == "rop5_inpt")
2091 {
2092     /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
2093     la posición 4, si es que son iguales el puntaje sube en 1*/
2094     if (textoBot.text == alternaJera[4])
2095     {
2096         puntajeFinal++;
2097     }
2098     /*Y en la posición 8 de mi arreglo de respuestasDadas agrego el valor ingresado + el tiempo de respuesta
2099     + la pregunta que se encuentre en la posición 4 del arreglo numerosGuardados +1*/
2100     respuestasDadas[8] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[4] + 1);
2101 }
2102 //si el nombre del botón es igual a rop5dias_inpt
2103 else if (textoBot.name == "rop5dias_inpt")
2104 {
2105     /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
2106     la posición 4, si es que son iguales el puntaje sube en 1*/
2107     if (textoBot.text == alternaDias[4])
2108     {
2109         puntajeFinal++;
2110     }
2111     /*Y en la posición 9 de mi arreglo de respuestasDadas agrego el valor ingresado + el tiempo de respuesta
2112     + la pregunta que se encuentre en la posición 4 del arreglo numerosGuardados +7*/
2113     respuestasDadas[9] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[4] + 7);
2114 }
2115 //si el nombre del botón es igual a rop6_inpt
2116 if (textoBot.name == "rop6_inpt")
2117 {
2118     /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
2119     la posición 5, si es que son iguales el puntaje sube en 1*/
2120     if (textoBot.text == alternaJera[5])
2121     {
2122         puntajeFinal++;
2123     }
2124     /*Y en la posición 10 de mi arreglo de respuestasDadas agrego el valor ingresado + el tiempo de respuesta
2125     + la pregunta que se encuentre en la posición 5 del arreglo numerosGuardados +1*/
2126     respuestasDadas[10] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[5] + 1);
2127 }
2128 //si el nombre del botón es igual a rop6dias_inpt
2129 else if (textoBot.name == "rop6dias_inpt")
2130 {
2131     /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
2132     la posición 5, si es que son iguales el puntaje sube en 1*/
2133     if (textoBot.text == alternaDias[5])
2134     {
2135         puntajeFinal++;
2136     }
2137     /*Y en la posición 11 de mi arreglo de respuestasDadas agrego el valor ingresado + el tiempo de respuesta
2138     + la pregunta que se encuentre en la posición 5 del arreglo numerosGuardados +7*/
2139     respuestasDadas[11] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[5] + 7);
2140 }
2141 }
2142

```

Fig 230 "Fragmento de código de la función Respondo() parte 3"

```

2143 public void Post()
2144 {
2145     var request = new UnityWebRequest("https://simulab.edutech-project.org/api/registrarActividad", "POST");
2146     string auxJSON = JsonUtility.ToJson(ejercicio);
2147     byte[] bodyRaw = Encoding.UTF8.GetBytes(auxJSON);
2148     request.uploadHandler = (UploadHandler)new UploadHandlerRaw(bodyRaw);
2149     request.downloadHandler = (DownloadHandler)new DownloadHandlerBuffer();
2150     request.SetRequestHeader("Content-Type", "application/json");
2151     request.SendWebRequest();
2152 }
2153
2154

```

Fig 231 "Funcion Post()"

```

2156     public static void MenuSel(string tipoMen)
2157     {
2158         GeneralEmpezar.menu = tipoMen;
2159     }

```

Fig 232 "Funcion MenuSel()"

```
2206 [Serializable]
2207 public class Pregunta
2208 {
2209     public string respuestaIngresada;
2210     public string tiempoRespuesta;
2211     public int numeroPregunta;
2212     public DateTime inicio;
2213     public DateTime fin;
2214     public string getRespuetaIngresada()
2215     {
2216         return respuestaIngresada;
2217     }
2218     public string getTiempoRespuesta()
2219     {
2220         return tiempoRespuesta;
2221     }
2222     public void setNumeroPregunta(int numeroPregunta)
2223     {
2224         this.numeroPregunta = numeroPregunta;
2225     }
2226     public int getPregunta()
2227     {
2228         return numeroPregunta;
2229     }
2230     public void setInicio()
2231     {
2232         this.inicio = System.DateTime.Now;
2233     }
2234     public void setFin()
2235     {
2236         this.fin = System.DateTime.Now;
2237     }
2238     public void setTiempoRespuesta(string tiempoRespuesta)
2239     {
2240         this.tiempoRespuesta = tiempoRespuesta;
2241     }
2242     public void setRespuetaIngresada(string respt)
2243     {
2244         this.respuestaIngresada = respt;
2245     }
2246 }
```

Fig 233 "Clase Pregunta"

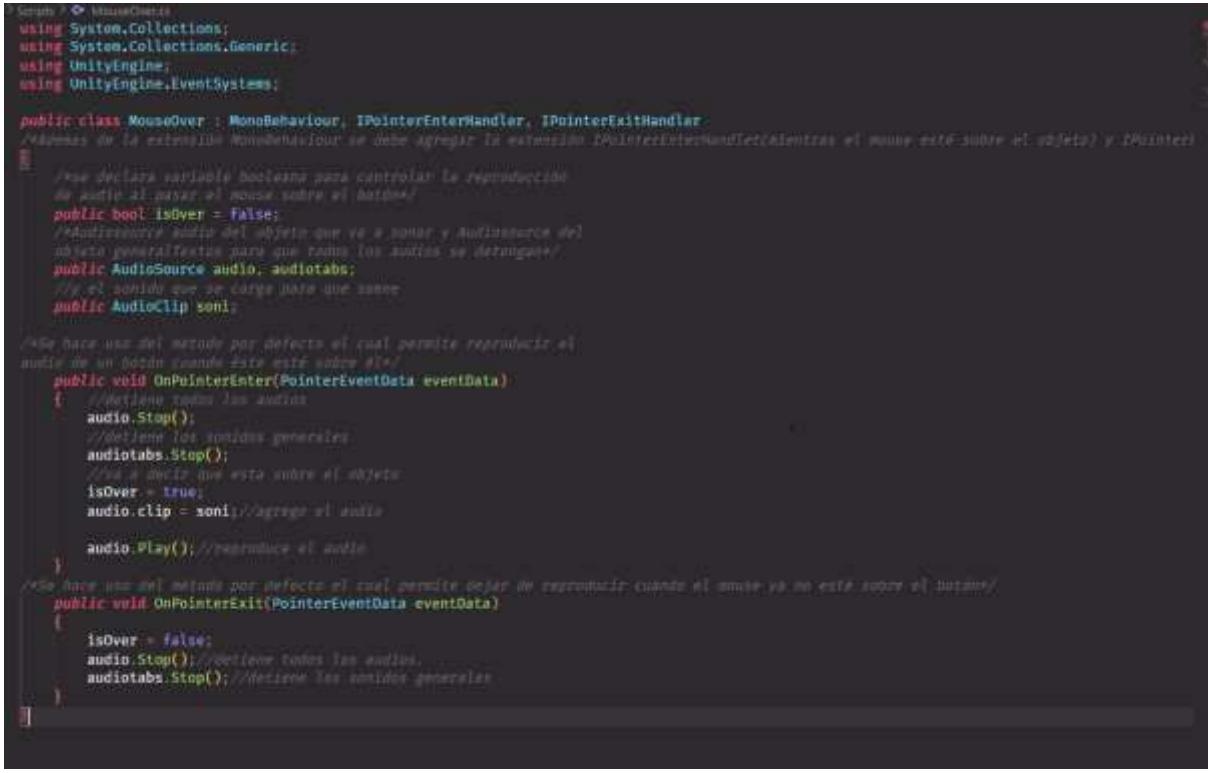
```
[Serializable]
public class SimuladorData
{
    public string correo;
    public int numeroEjercitario;
    public string tiempoInicio;
    public string tiempoFin;
    public string fechaDeActividad;
    public List<Pregunta> preguntas = new List<Pregunta>();
    public void ReinicioPreguntas()
    {
        preguntas = new List<Pregunta>();
    }
    public void setNumeroEjercitario(int numero)
    {
        this.numeroEjercitario = numero;
    }
    public int getNumeroEjercitario()
    {
        return this.numeroEjercitario;
    }
    public void setTiempoInicio(string tiempoInicio)
    {
        this.tiempoInicio = tiempoInicio;
    }
    public string getTiempoInicio()
    {
        return this.tiempoInicio;
    }
    public void setTiempoFin(string tiempoFin)
    {
        this.tiempoFin = tiempoFin;
    }
    public string getTiempoFin()
    {
        return this.tiempoFin;
    }
    public void setFechaActividad(string fechaDeActividad)
    {
        this.fechaDeActividad = fechaDeActividad;
    }
    public string getFechaActividad()
    {
        return this.fechaDeActividad;
    }
    public void setCorreo(string correo)
    {
        this.correo = correo;
    }
    public string getCorreo()
    {
        return this.correo;
    }
    public void addPregunta(Pregunta nuevaPreg)
    {
        this.preguntas.Add(nuevaPreg);
    }
    public void updatePregunta(Pregunta upPreg, Pregunta antPreg)
    {
        this.preguntas.Remove(antPreg);
        this.preguntas.Add(upPreg);
    }
    public Pregunta readPregunta(int indice)
    {
        return preguntas[indice];
    }
    public void setInicioPregunta(int numero)
    {
        this.preguntas[numero].setInicio();
    }
}
```

Fig 234 "Clase SimuladorData"

```
{
  "correo": "jbarrerab1@est.ups.edu.ec",
  "numeroEjercitario": 5,
  "tiempoInicio": "09:24:58",
  "tiempoFin": "09:25:47",
  "fechaDeActividad": "2022-04-06 09:25:47",
  "preguntas": [
    {"respuestaIngresada": "1", "tiempoRespuesta": "00:23:08", "numeroPregunta": 3},
    {"respuestaIngresada": "5", "tiempoRespuesta": "00:01:63", "numeroPregunta": 9},
    {"respuestaIngresada": "3", "tiempoRespuesta": "00:01:05", "numeroPregunta": 2},
    {"respuestaIngresada": "7", "tiempoRespuesta": "00:00:45", "numeroPregunta": 8},
    {"respuestaIngresada": "6", "tiempoRespuesta": "00:01:75", "numeroPregunta": 6},
    {"respuestaIngresada": "10", "tiempoRespuesta": "00:01:84", "numeroPregunta": 12},
    {"respuestaIngresada": "1", "tiempoRespuesta": "00:01:94", "numeroPregunta": 1},
    {"respuestaIngresada": "1", "tiempoRespuesta": "00:01:51", "numeroPregunta": 7},
    {"respuestaIngresada": "1", "tiempoRespuesta": "00:00:42", "numeroPregunta": 4},
    {"respuestaIngresada": "1", "tiempoRespuesta": "00:01:13", "numeroPregunta": 10},
    {"respuestaIngresada": "1", "tiempoRespuesta": "00:01:68", "numeroPregunta": 5},
    {"respuestaIngresada": "1", "tiempoRespuesta": "00:01:95", "numeroPregunta": 11}
  ]
}
```

Fig 235"Esstructura de preguntas que se genera en el archivo json"

### 1.5.2. Script MouseOver



```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Events;

public class MouseOver : MonoBehaviour, IPointerEnterHandler, IPointerExitHandler
{
  //Añades de la extensión MonoBehaviour se debe agregar la extensiones IPointerEnterHandler(entre el mouse este sobre el objeto) y IPointerExitHandler(entre el mouse se quita el mouse del objeto)
  //esta variable booleana para controlar la reproducción de audio al pasar el mouse sobre el botón
  public bool isOver = false;
  //audio source del objeto que va a sonar y AudioSource del objeto general para que tanto los audios se detengano
  public AudioSource audio, audiotabs;
  //el sonido que se carga para que suene
  public AudioClip soni;

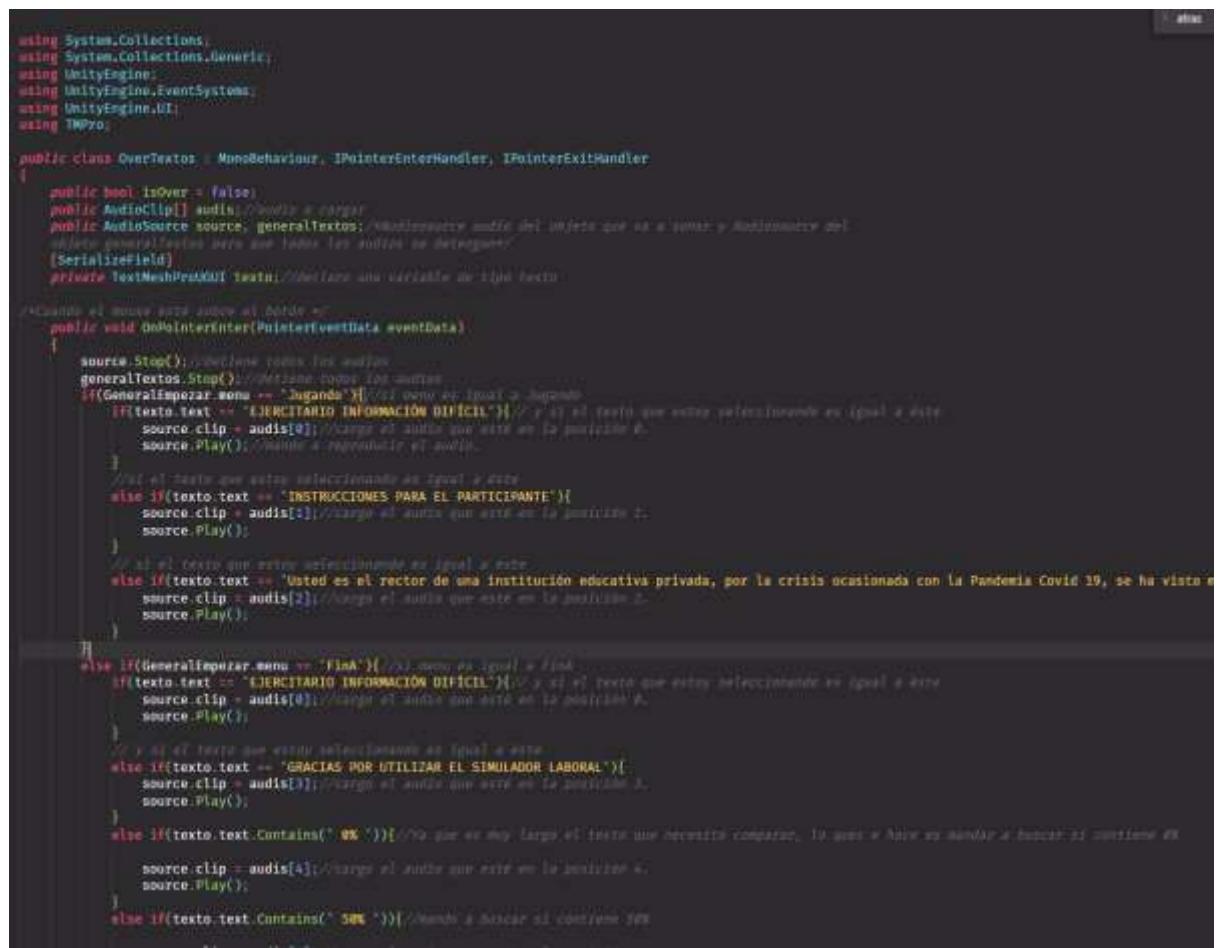
  //Se hace uso del metodo por defecto el cual permite reproducir el audio con un boton cuando este este sobre el
  public void OnPointerEnter(PointerEventData eventData)
  {
    //detiene todos los audios
    audio.Stop();
    //detiene los sonidos generales
    audiotabs.Stop();
    //se indica que esta sobre el objeto
    isOver = true;
    audio.clip = soni; //agrega el audio
    audio.Play(); //reproduce el audio
  }

  //Se hace uso del metodo por defecto el cual permite dejar de reproducir cuando el mouse ya no este sobre el boton
  public void OnPointerExit(PointerEventData eventData)
  {
    isOver = false;
    audio.Stop(); //detiene todos los audios
    audiotabs.Stop(); //detiene los sonidos generales
  }
}

```

Fig 236"Fragmento de código de funciones para reproducir audio cuando el mouse está sobre un botón"

### 1.5.3. Script OverTextos.cs



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine;
using UnityEngine.EventSystems;
using UnityEngine.UI;
using TMPro;

public class OverTextos : MonoBehaviour, IPointerEnterHandler, IPointerExitHandler
{
    public bool isOver = false;
    public AudioClip[] audis;
    AudioSource source;
    generaltextos generaltextos; //Almacena audio del objeto que se oírás y Redimensiona el objeto generaltextos para que todos los audios se detenggan
    [SerializeField]
    private TextMeshProUGUI texto; //Almacena una variable de tipo texto

    //Cuando el mouse entra sobre el botón
    public void OnPointerEnter(PointerEventData eventData)
    {
        source.Stop(); //Almacena todos los audios
        generaltextos.Stop(); //Almacena todos los textos
        if(GeneralImpezar.menu == "Jugando") //Si estoy en modo Juego
        {
            if(texto.text == "EJERCITARIO INFORMACIÓN DIFÍCIL") //Y si el texto que estoy seleccionando es igual a este
                source.clip = audis[0]; //Carga el audio que está en la posición 0.
            source.Play(); //Manda a reproducir el audio
        }
        else if(texto.text == "INSTRUCCIONES PARA EL PARTICIPANTE")
        {
            source.clip = audis[1]; //Carga el audio que está en la posición 1.
            source.Play();
        }
        else if(texto.text == "Usted es el rector de una institución educativa privada, por la crisis ocasionada con la Pandemia Covid 19, se ha visto en la obligación de cancelar las clases presenciales y solo impartir clases virtuales. Usted tiene que pensar en cómo manejar la situación y qué estrategias implementar para garantizar la continuidad de la educación en su institución." )
        {
            source.clip = audis[2]; //Carga el audio que está en la posición 2.
            source.Play();
        }
        else if(GeneralImpezar.menu == "Final")
        {
            if(texto.text == "EJERCITARIO INFORMACIÓN DIFÍCIL") //Y si el texto que estoy seleccionando es igual a este
                source.clip = audis[0]; //Carga el audio que está en la posición 0.
            source.Play();
        }
        else if(texto.text == "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL")
        {
            source.clip = audis[3]; //Carga el audio que está en la posición 3.
            source.Play();
        }
        else if(texto.text.Contains("ex"))
        {
            source.clip = audis[4]; //Carga el audio que está en la posición 4.
            source.Play();
        }
        else if(texto.text.Contains("se"))
        {
            source.clip = audis[5]; //Carga el audio que está en la posición 5.
            source.Play();
        }
    }

    //Cuando el mouse sale sobre el botón
    public void OnPointerExit(PointerEventData eventData)
    {
    }
}
```

Fig 237"Fragmento de código del script OverTextos.cs"

### 1.5.4. Script MouseColor.cs

```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Events;
using UnityEngine.UI;

public class MouseColor : MonoBehaviour, IPointerEnterHandler, IPointerExitHandler
{
    //Añadiremos esta variable en Value, para que solo se active cuando pasa el mouse sobre el objeto
    public bool isOver = false;
    public Image botonAct; //Añadiremos una variable en tipo Image

    //Esta función se activa cuando el mouse pasa sobre el objeto
    public void OnPointerEnter(PointerEventData eventData)
    {
        isOver = true;
        //Si el botón es blanco
        if(Preferscript.colorBt == "Blanco"){
            //Asigna la propiedad color del botón
            var temColor = botonAct.color;
            temColor.a = 0.5f;
            temColor.r = 0.52f;
            temColor.g = 0.5f;
            temColor.b = 0.5f;
            botonAct.color = temColor; //Le asigna un nuevo color
        }
        //Si el mouse del botón es rojo
        else if(botonAct.name == "Teal1"){
            var temColor = botonAct.color;
            temColor.a = 0.5f;
            temColor.r = 0.52f;
            temColor.g = 0.5f;
            temColor.b = 0.5f;
            botonAct.color = temColor; //Le asigna un nuevo color
        }
        else{
            //Asigna la propiedad color del botón
            var temColor = botonAct.color;
            temColor.a = 0.75f;
            temColor.r = 0.52f;
            temColor.g = 0.5f;
            temColor.b = 0.5f;
            botonAct.color = temColor;
        }
    }
}

```

*Fig 238"Función que al estar el mouse sobre un botón cambia de color"*

```

//La función recibe como parámetro el mouse que está dentro
public void OnPointerExit(PointerEventData eventData)
{
    isOver = false;
    //Si el mouse que entra es rojo, cambia el color
    if(botonAct.name == "10" || botonAct.name == "31" || botonAct.name == "Dyslexic" || botonAct.name == "Late" || botonAct.name == "Arial Bold"){
        if(Preferscript.colorBt == "Blanco")//Si el botón es blanco
            botonAct.color = Color.white; //Le asigna un nuevo color
    }
    else{
        //Le asigna un nuevo color
        botonAct.color = Color.black;
    }
}

//Si el mouse del botón contiene _act
else if(botonAct.name.Contains("_act")){
    if(Preferscript.colorBt == "Blanco" || Preferscript.colorBt == "Negro")
        botonAct.color = Color.white;
    else
        botonAct.color = Color.black;
}

```

*Fig 239"Función que al salir el mouse del botón cambia al color por defecto"*

### 1.5.5. Script PreferSctipt.cs

```
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  using TMPro;
5  using UnityEngine.UI;
6  using System.IO;
7
8  public class PreferSctipt : MonoBehaviour
9  {
10     [SerializeField]
11     private Image[] botones; //Arreglo de imágenes utilizadas para cambiar el color de los botones.
12     [SerializeField]
13     //Arreglo de textos utilizados para cambiar el color del texto.
14     cambio_fuente = texto_y_cambio_de_tamaño();
15     private TextMeshProUGUI[] objeto_preg; objPreg;
16     [SerializeField]
17     //Arreglo de fuentes al cual contiene las fuentes a utilizar.
18     private TextMeshProUGUI[] tiposFuente;
19     [SerializeField]
20     //Arreglo de textos el cual contiene los títulos de contenidos a usar.
21     private TextMeshProUGUI tem1, tem2;
22     [SerializeField]
23     //Arreglo de imágenes el cual contiene el panel,
24     el botón de desactivar audio y el panel de preferencias.
25     private Image panel, muteBtm, preff;
26     //Variable de tipo button
27     public Button botMute;
28     public AudioSource[] sonidoAS;
29     //Variable de tipo texto del botón silenciar.
30     public TextMeshProUGUI silbtm;
31     //Variable existente para controlar en que menú estamos.
32     public static string menu="Main";
33     //Variable existente para controlar el tipo de contraste
34     public static string colorRt="Negro";
35     //Variable existente para controlar si audio.
36     public static bool isMute = false;
37     [SerializeField]
38     //se crea un arreglo de imágenes de los botones principales.
39     private Image[] botonesImagenes;
40     //se crea una variable para controlar el activar y desactivar el audio.
41     public Sprite ImagenMute, ImagenMuteS;
42     //variable booleana para controlar cuando la función Rolyxie está activa.
43     private bool Disle=false;
44
45 }
```

Fig 240 "Declaración de variables"

```
46     public void Cambio(string colpan){
47         if (colpan == "Blanco"){//si el string que recibe es igual a Blanco
48             panel.color = new Color(255,255,255,100); //cambiamos el color del panel a blanco
49             preff.color=new Color(50,50,50,100); //cambiamos el color del panel de preferencias a oscuro.
50         } //si el string que recibe es igual a Negro
51         else if (colpan == "Negro"){
52             panel.color = new Color(0,0,0,215); //cambiamos el color del panel a negro
53             preff.color=Color.grey; //cambiamos el color del panel de preferencias a gris.
54         }
55     }
56 }
```

Fig 241 "Función para cambiar el color del panel"

```
57 /*Funcion que permite silenciar la salida de audio del simulador*/
58 public void MuteAll(){
59     //si la variable estatica es igual a false y colorBt es blanco
60     if(isMute == false && PreferSctipt.colorBt == "Blanco"){
61         //mandamos a silenciar todos los audios.
62         foreach (var s in sonidoAS){
63             s.volume = 0f;
64         }
65         isMute = true;
66         //si esta desactivado el audio asignamos la imagen ImagenMute al botón
67         botMute.image.sprite = ImagenMute;
68         muteBtn.color = Color.yellow;//cambiamos a amarillo
69     }
70     //si la variable estatica es igual a false y colorBt es Negro
71     else if(isMute == false && PreferSctipt.colorBt == "Negro"){
72         //mandamos a silenciar todos los audios.
73         foreach (var s in sonidoAS){
74             s.volume = 0f;
75         }
76         isMute = true;
77         botMute.image.sprite = ImagenMute;
78         muteBtn.color = Color.white;
79     }
80     else { //caso contrario
81         //mandamos a activar la salida de audios.
82         foreach (var s in sonidoAS){
83             s.volume = 1f;
84         }
85         isMute = false;
86         //Si colorBt es blanco
87         if(PreferSctipt.colorBt == "Blanco"){
88             //cambiamos la imagen
89             botMute.image.sprite = ImagenMuteS;
90             //cambiamos a blanco
91             muteBtn.color = Color.white;
92         }
93         else{
94             botMute.image.sprite = ImagenMuteS;
95             //caso contrario lo cambia a amarillo
96             muteBtn.color = Color.yellow;
97         }
98     }
99 }
100 }
```

Fig 242 "Función para desactivar todos los audios"

```

1  /*Función que recibe como parametro un string dicha función permite cambiar el color de los botones*/
2  public void CambioColBot(string te){
3      //si el string que recibe es igual a Blanco
4      if (te == "Blanco"){
5          //cambia todos los botones a blanco
6          foreach(var b in botones){
7              b.color = Color.white;
8              PreferSctipt.ColB("Blanco");
9          }
10         foreach(var b in botonesImagenes){
11             b.color = Color.white;
12         }
13         foreach(var b in pregis){
14             b.color = Color.white;
15         }
16     }
17     //si el string que recibe es igual a Negro
18     else if (te == "Negro"){
19         //cambia todos los botones a negro
20         foreach(var b in botones){
21             b.color = Color.black;
22             PreferSctipt.ColB("Negro");
23         }
24         //los botones principales los cambiamos a amarillo
25         foreach(var b in botonesImagenes){
26             b.color = Color.yellow;
27         }
28     }
29 }

```

Fig 243 "Función para cambiar el color de los botones"

```

131 /*Esta función recibe como parametro un string dicha función permite cambiar el color del texto*/
132 public void CambioColText(string te){
133     //si el string que recibe es igual a Negro
134     if (te == "Negro"){
135         /*cambia la letra a azul*/
136         foreach (var obj in objeto) {
137             obj.color= Color.blue;
138         }
139         foreach (var obj in tiposLetra) {
140             obj.color= Color.blue;
141         }
142         foreach (var obj in pregis) {
143             obj.color= Color.blue;
144         }
145         foreach (var obj in objPrefer) {
146             obj.color= Color.blue;
147         }
148     }
149     //si el string que recibe es igual a Amarillo
150     if (te == "Amarillo"){
151         /*cambia la letra a amarillo*/
152         foreach (var obj in objeto) {
153             obj.color = Color.yellow;
154         }
155         foreach (var obj in tiposLetra) {
156             obj.color = Color.yellow;
157         }
158         foreach (var obj in pregis) {
159             obj.color= Color.yellow;
160         }
161         foreach (var obj in objPrefer) {
162             obj.color=Color.yellow;
163         }
164     }
165 }

```

Fig 244 "Función para cambiar el color del texto de todos los componentes de la interfaz"

```
168 /*Esta función permite recibir como parámetro un string al cual se le pasa la fuente, esta función  
169 permite personalizar el texto de la interfaz haciendo uso de tres tipos de fuente de texto*/  
170 public void cambioFuente(TMP_FontAsset fu){  
171     //si la fuente que recibe es igual a  
172     if(fu.ToString() == "OpenDyslexic-Regular SDF (TMPPro.TMP_FontAsset)"){  
173         //se define un tamaño para cuando esté activa esta fuente.*/  
174         foreach(var t in objeto){  
175             t.fontSize = 19f;  
176         }  
177         foreach(var t in pregis){  
178             t.fontSize = (18f);  
179         }  
180         foreach(var t in objPrefer){  
181             t.fontSize = (17f);  
182         }  
183  
184         Disle=true;//si está activa  
185     }  
186     else{  
187         Disle=false;// si esta desactivada  
188     }  
189     /*cambia de fuente a todos los textos de la interfaz*/  
190     foreach(var t in objeto){  
191         t.font = fu;  
192     }  
193     foreach(var t in pregis){  
194         t.font = fu;  
195     }  
196     foreach(var t in objPrefer){  
197         t.font = fu;  
198     }  
199     //cambia el tipo de fuente  
200     tem1.font = fu;  
201     tem2.font = fu;  
202 }  
203 }
```

Fig 245 "Función para cambiar la fuente texto de toda la interfaz"

```

204 /*Función que recibe como parámetro un entero dicha función permite reducir y aumentar el tamaño de texto*/
205 public void cambioTamano(int tam){
206
207     if(Disle=true){//si esta activa dicha fuente
208         /*del tamaño definido restamos -2 */
209         foreach(var t in objeto){
210             t.fontSize = tam-2;
211         }
212         /*del tamaño definido restamos -7 a objetos de preferencias*/
213         foreach(var t in objPrefer){
214             t.fontSize = tam-7;
215         }
216         /*del tamaño definido restamos -7 a preguntas */
217         foreach(var t in pregts){
218             t.fontSize = (tam-7);
219         }
220         //cambia el tamaño
221         tem1.fontSize = tam;
222         tem2.fontSize = tam;
223     }
224
225     else{//si Disle es igual a false
226         /*cambiamos el tamaño de texto a todos los objetos*/
227         foreach(var t in objeto){
228             t.fontSize = tam;
229         }
230         foreach(var t in pregts){
231             t.fontSize = (tam);
232         }
233         foreach(var t in objPrefer){
234             t.fontSize = tam;
235         }
236         //cambia el tamaño
237         tem1.fontSize = tam;
238         tem2.fontSize = tam;
239     }
240 }
241 /*Función que permite detener todos los sonidos cuando se desactive el audio */

```

Fig 246"Función para cambiar el tamaño de texto"

```

228     public static void MenuSel(string tipoMen){
229         Debug.Log("Eras "+PreferSctipt.menu+" ahora eres "+tipoMen);
230         PreferSctipt.menu = tipoMen;
231     }
232

```

Fig 247"Función para controlar en que panel se posiciona"

```

233     public static void ColB(string col){
234         PreferSctipt.colorBt = col;
235     }

```

Fig 248"Función que controla el cambio de color de los botones según el color del panel que se vaya a definir"

#### 1.5.6. Script WebData.cs

```

using UnityEngine;
using System.Runtime.InteropServices;
using UnityEngine.UI;

public class WebData : MonoBehaviour {

    [DllImport("_Internal")]
    private static extern int GetUrlVal();

    [DllImport("_Internal")]
    private static extern int GetEjer();

    [DllImport("_Internal")]
    private static extern string GetMail();

    public Text idVal;
    public Text idMail;
    public Text idEjer;

    void Start() {
    }

    public int ID(){
        int id = GetUrlVal();
        Debug.Log("ID: "+id);
        idVal.text = " "+id;
        return id;
    }

    public string Mail(){
        string mailV = GetMail();
        Debug.Log("Mail - Un: "+mailV);
        idMail.text = " "+mailV;
        return mailV;
    }

    public int Ejercitario(){
        int ejerV = GetEjer();
        Debug.Log("Ejer - Un: "+ejerV);
        idEjer.text = " "+ejerV;
        return ejerV;
    }
}

```

Fig 249 "Script WebData para los servicios web"

## 1.6. Directorio del proyecto de cada Simulador Laboral

- 1) Carpeta que contiene las animaciones creadas para el proyecto.
- 2) Carpeta que contiene los audios utilizados para la descripción de textos.
- 3) Carpeta que contiene las imágenes utilizadas para los botones.
- 4) Carpeta que contiene imágenes utilizadas para animar objetos del escenario(laptop).
- 5) Carpeta que contiene las fuentes de texto utilizadas.
- 6) Carpeta que contiene materiales utilizados en el escenario.
- 7) Carpeta que contiene al personaje del escenario.
- 8) Carpeta que contiene el archivo para la comunicación de la API de Unity con el servidor web.
- 9) Carpeta que contiene la escena del proyecto.
- 10) Carpeta que contiene los scripts generados para el proyecto.



Fig 250 "Directorio del proyecto"

## 1.7 Construyendo y ejecutando los simuladores laborales en WebGL

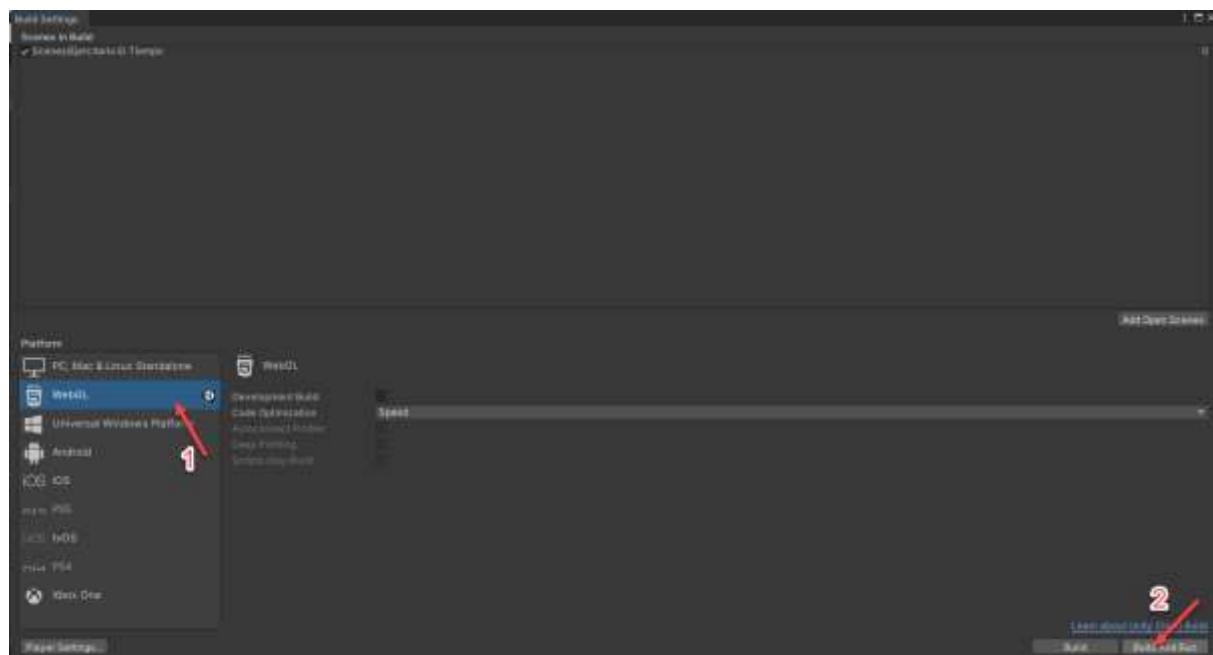


Fig 251 "Construcción a WebGL"



Fig 252 "Ejecutando en WebGL Simulador Información Difícil"

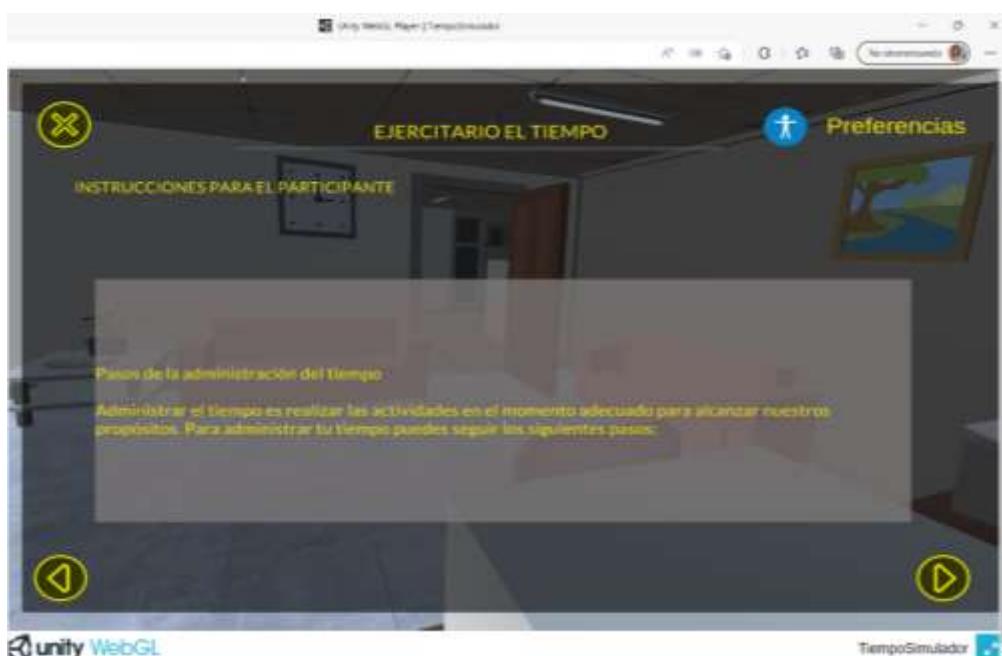


Fig 253 "Simulador Laboral El Tiempo"