Materia: **Taller de Programación 2**. Escuela ORT sede Almagro
Año: 4to Informática

TP5 – Sala de Escape

Desarrollar un juego de escape que podés terminar, con todas las claves que abren las puertas

Atención: No programar hasta que la temática, las pistas y las imágenes estén hechas!

Objetivos del tp

Utilizar el valor de los campos completados en formularios Web

Terminar de familiarizarse con los IActionResult del Controller, el desarrollo de vistas y la creación de models.

Crear un proyecto de ASP.NET Core MVC llamado **SalaDeEscape**

Models

Crear la clase estática **Escape**.cs

Debe contener los siguientes atributos:

string[] **_incognitasSalas**

(Aquí almacenaremos las soluciones de cada sala en formato texto. En el caso de que un resultado de sala sea en otro tipo de dato convertirlo a string antes de compararlo)

int **_estadoJuego**;

(Indicara que numero de Sala fue la última resuelta, Inicialmente en 1)

La siguiente Propiedad de lectura solamente:

int **EstadoJuego**

Y los siguientes Métodos:

private void **InicializarJuego()**

Inicializa el juego con el array de incógnitas correctas.

public bool **ResolverSala**(int Sala, string Incognita)

(Recibe el número de sala a resolver y la incógnita elegida por el

usuario. Debe validar con _EstadoJuego si puede resolver dicha sala (devolver false si está resolviendo una sala a la que aún no tenga acceso) y luego validar la incógnita de la sala en cuestión devolviendo true o false si es correcta o no.) incrementando el _estadojuego en caso de ser correcta.

Tener en cuenta que si el array de incógnitas tiene 0 elementos debe llamar a InicializarJuego (como ocurrió con el TP anterior)

Controller

HomeController

Action Methods:

Index: Retorna la View Index con la presentación del juego.

Tutorial: Retorna la View Tutorial con la explicación del juego.

Comenzar → Devuelve la view de la próxima Habitación a resolver (Según lo que indique EstadoJuego)

***[HttpPost]* Habitacion**(int sala, string clave) → Verifica que la sala que se está respondiendo coincida con EstadoJuego. En caso de estar resolviendo una sala incorrecta deberá devolver la view de la sala a resolver.

Luego Verifica que la clave ingresada sea la correcta. De ser así, retorna la View siguiente Habitacion. De lo contrario, vuelve a la misma vista y llena el ViewBag.Error informando al usuario que la respuesta escrita fue incorrecta.

En caso de estar resolviendo la última sala de manera correcta deberá ir a la View Victoria.

Views

Masterpage

_Layout.cshtml

El contenido del **body** debe ser:

navbar (barra de navegación) fijada en la parte superior con:

Nombre del juego (link a la página principal del sitio)

Tutorial (link a una acción llamada tutorial)

Jugar (link a la acción Comenzar)

footer en el cual figura el nombre y apellido del autores/as del sitio, Copyright y año.

index.cshtml

Debe contener: Nombre, Logo del juego y slogan. Luego debe haber 3 imágenes descriptivas del juego)

Quien ya tenga noción de Bootstrap puede pensar en hacer un carousel para las imágenes descriptivas.

tutorial.cshtml

Debe incluir una explicación con texto e imágenes acerca del juego, separado en:

Breve reseña explicando de qué trata la temática elegida.

Imágenes de las distintas escenas del juego

El diseño lo dejamos a tu criterio.

habitacionX.cshtml (X Refiere a las vistas habitacion1, habitacion2, etc…)

Todas las view de las habitaciones tienen los mismos elementos (con distintos valores) :

Un título de la habitación

Un texto relatando lo que se encuentra en la sala y lo que hay que hacer

Un elemento o varios elementos multimedia (por ejemplo una imagen, GIF, un video embebido de Youtube con iframe, etc) que de a entender la clave a descifrar

En las 4 habitaciones deberás utilizar un campo input con texto para lograr descifrar: (en ningún caso se permiten controles Checkbox, Radio y Select)

El botón “VALIDAR” que envía el form (POST, con action a la habitación)

2 botones que dan pistas para adivinar la clave.

Podemos mostrar mensajes emergentes para las pistas, de la siguiente forma:

<button **onclick="alert('Pista 1: Mirá las paredes...')**"> Pista 1 </button> o bien utilizando el control Modal de Bootstrap.

victoria.cshtml

El contenido y diseño de esta view queda a tu criterio. Demostremos nuestro entusiasmo al usuario que logró salir correctamente de la sala de escape.

Aclaraciones importantes

Podés tener como referencia a la hora de armar tu experiencia (y sólo como referencia, no queremos una copia, sino que le pongas tu propia impronta), este juego de sala de escape

Formato de entrega

Subir la carpeta completa del proyecto en un archivo comprimido al servicio de entrega del Campus Virtual con el nombre de archivo “TP5_Binker_Kristal_Medina.zip”

Extras para quienes van terminando...

Crear un juego que permita la participación e interacción de 2 jugadores (tal cual mostramos en el ejemplo guía)

Mostrar en la view victoria.cshtml, la cantidad de intentos extra y pistas extra que necesitó el usuario para adivinar la clave (utilizar la clase estática Escape para almacenar dicha información)

En la vista index de escape, pedir como ingreso el nombre del jugador, y mostrar ese valor en todas las vistas dentro del juego (definir un espacio para alojarlo en la navbar). Además, mostrar en victoria.cshtml, el cartel informando al usuario que ganó, con el nombre del mismo.

(utilizar la clase estática Escape para almacenar dicha información)

Un Súper Desafío: Implementar un Timer con el tiempo que lleva el usuario dentro de la sala, y visualizarlo en todas las vistas (Máximo 1 hora).