Universidad Tecnológica Nacional Facultad Regional Avellaneda										
Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos										
Materia: Laboratorio de Programación II										
Apellido:			Fecha:		21/06/2022					
Nombre:			Docente(2):							
División:	2ºC		Nota(2):							
Legajo:			Firma(2):							
Instancia(1):	**PP**		**RPP**		**SP**	X	**RSP**		**FIN**	

(1) Las instancias validas son: 1er Parcial (**PP**), Recuperatorio 1er Parcial (**RPP**), 2do Parcial (**SP**), Recuperatorio 2do Parcial (**RSP**), Final (**FIN**). Marque con una cruz.
(2) Campos a ser completados por el docente.

IMPORTANTE:

- **2 (dos) errores en el mismo tema anulan su puntaje.**
- La correcta documentación y reglas de estilo de la cátedra serán evaluadas.
- El proyecto debe ser creado en .Net 5.
- Colocar sus datos personales en el nombre de la carpeta principal y la solución: Apellido.Nombre.Div. Ej: Pérez.Juan.2D. No sé corregirán proyectos que no sea identificable su autor.
- No se corregirán exámenes que no compilen.
- **Reutilizar** tanto código como crean necesario.
- Colocar nombre de la clase (en estáticos), **this** o **base** en todos los casos que corresponda.
- Aplicar los principios de los 4 pilares de la POO.

---

*TIEMPO MÁXIMO PARA RESOLVER EL EXAMEN **90** MINUTOS.*

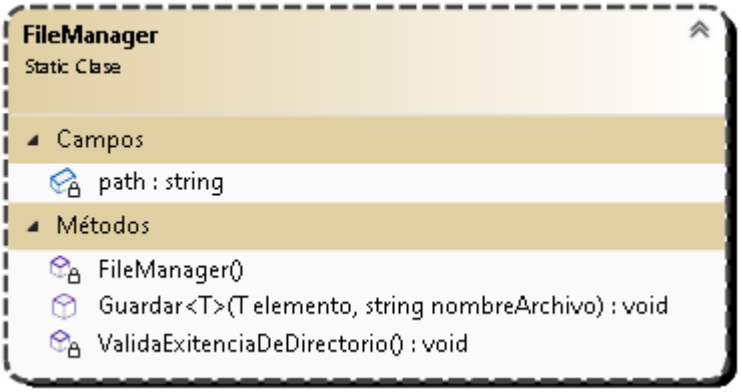
---

1. Partir de la solución entregada. Modificar su nombre con el siguiente formato: [APELLIDO].[NOMBRE].
2. Implementar la BD desde el backup enviado.

20220621SP.bak

Files

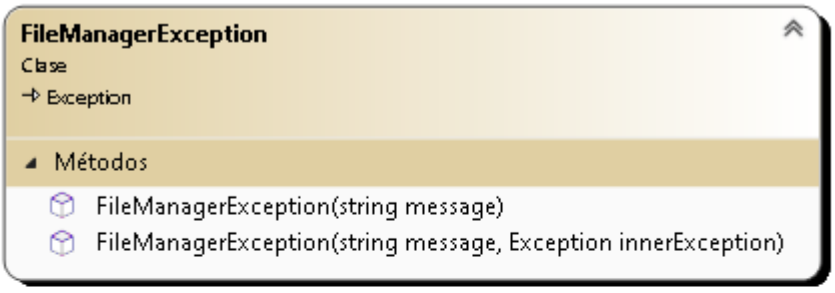
3. Dentro del proyecto se deberá respetar el siguiente esquema:

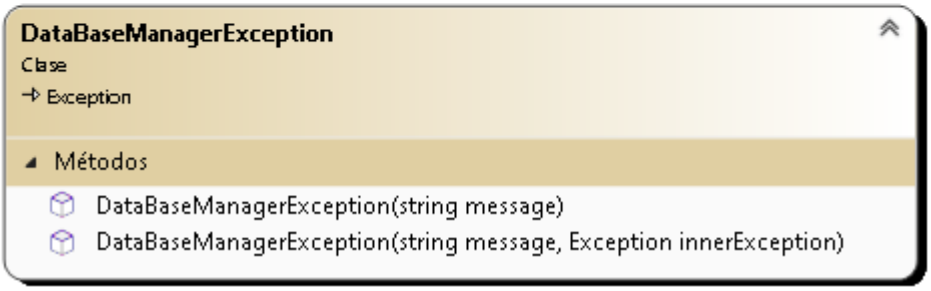


4. FileManager será estática.
    a. En el constructor de clase realizar:
        i. En el atributo path se almacenará la referencia al escritorio de la pc. Y se le concatenara un el nombre de la carpeta del parcial: ej {path escritorio}+`\\20220621SP\\`
        ii. Llamar al método ValidaExistenciaDeDirectorio.
    b. ValidaExistenciaDeDirectorio:
        i. Si no existe el directorio almacenado en path, se creará.
        ii. En caso de producirse una excepción al momento de la creación, esta deberá ser capturada y relanzada en una nueva excepción denominada FileManagerException, la cual contendrá el mensaje: “Error al crear el directorio”.
    c. Guardar:
        i. Será genérico y solo permitirá que los elementos a almacenar sean tipos por referencia.
        ii. Validar la extensión del nombre del archivo. En caso de que sea:
            1. JSON, se serializará el elemento recibido.
            2. TXT, se almacena en texto plano.
            3. Cualquier otra extensión se lanzará una excepción denominada FileManagerException, la cual contendrá el mensaje “Extensión no permitida”.

Excepciones

5. Dentro del proyecto respetar el siguiente esquema:

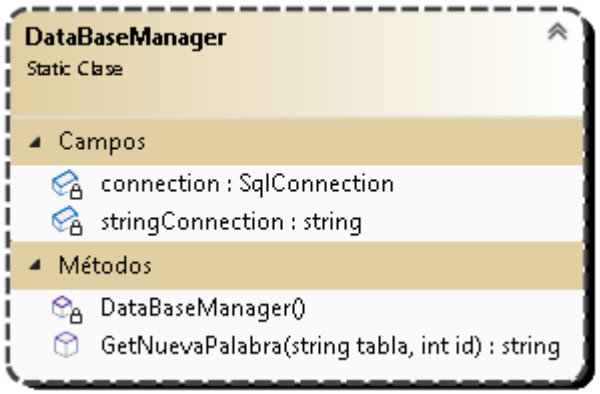




6. Controlar las posibles excepciones producidas e informar al usuario del error.

Bases de datos

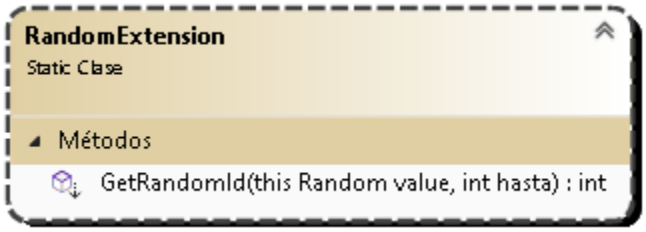
7. Dentro del proyecto se deberá respetar el siguiente esquema:



8. DataBaseManager será estática:
    a. En el constructor de clase inicializar el string connection.
    b. GetNuevaPalabra, recibirá el nombre de la tabla sobre la cual realizar el select y el id de la palabra a obtener. Retornada la palabra leída desde la BD.

Métodos de extensión

9. Dentro del proyecto se deberá respetar el siguiente esquema:



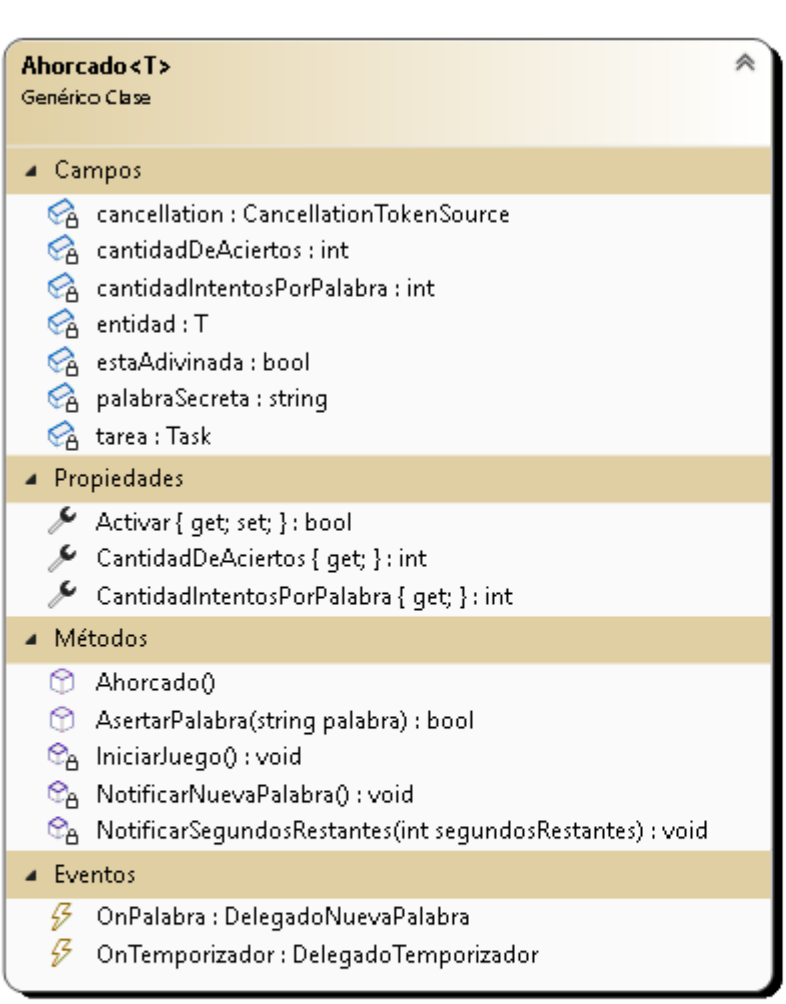
10. Extenderá la clase Random la cual retornará un valor de Id aleatorio desde 1 hasta el valor recibido por parámetro.

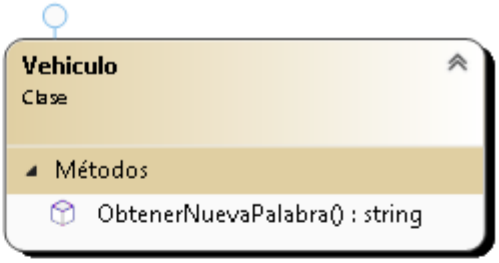
Interfaces

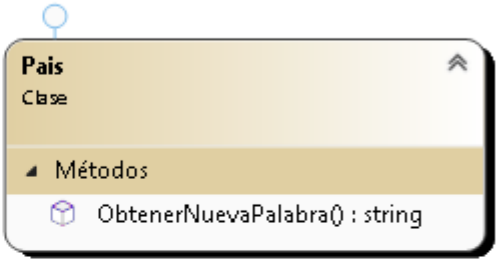
11. Dentro del proyecto se deberá respetar el siguiente esquema:

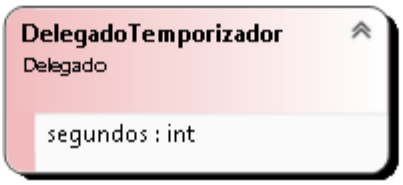
Entidades

12. Dentro del proyecto se deberá respetar el siguiente esquema:









13. **Vehículo**, implementará el mensaje ObtenerNuevaPalabra, para ello deberá leer desde la tabla “Vehiculos” en base a un ID aleatorio (hasta 37). Reutilizar código.
14. **País**, implementará el mensaje ObtenerNuevaPalabra, para ello deberá leer desde la tabla “Paises” en base a un ID aleatorio (hasta 35). Reutilizar código.
15. **Ahorcado**, será genérica, solo podrá recibir tipos que implementen la interfaz **Ilector** y posean un constructor publico sin parámetros:
    a. En su constructor publico sin parámetros realizar:
        i. Instanciar el atributo entidad.
        ii. Inicializar:
            1. estaAdivinada en false.
            2. cantidadIntentosPorPalabra y cantidadAciertos en 0 (cero).
            3. palabraSecreta en empty.
16. La propiedad Activar:
    a. El GET retornara True, si la tares no es nula y estado de la tarea es Running o WaitingToRun o WaitingForActivation.
    b. En el SET, si el valor recibido es TRUE y la tarea es nula o su estado no es Running o no es WaitingToRun o no es WaitingForActivation, se instanciará un nuevo CancelationTokenSource y se llamará a IniciarJuego. De lo contrario se llamará al método Cancel de cancellation.
17. El método IniciarJuego será privado y:
    a. Ejecutara en un hilo secundario la acción de que:

        i. Mientras no se requiera cancelación de la tarea invocara al mensaje NotificarNuevaPalabra y luego NotificarSegundosRestantes. Para este último enviar 30 segundos.
18. El método NotificarNuevaPalabra, verificara si el evento OnPalabra posee suscriptores y en caso exitoso realizara:
    a. Cambiar el estado del atributo estaAdivinada a False.
    b. Guardara en palabraSecreta el valor obtenido desde la entidad.
    c. cantidadDeIntentosPorPalabra será igual al doble de la longitud de la palabra secreta.
    d. Notificara la palabra secreta.
19. El método NotificarSegundosRestantes si posee un suscriptor notificara los segundos restantes mientras que (Utilizar Thread.Sleep para dormir el hilo 1 segundos antes de ir decrementando):
    a. segundosRestantes sea mayor o igual a cero.
    b. El hilo secundario no requiera cancelación.
    c. La palabra no haya sido adivinada.
    d. La cantidad de intentos sea mayor que 0 (cero).
20. El método AsertarPalabra comparara la palabra secreta con la recibida por parámetro (usar ToLower para comparar). Si son iguales cambiara el estado de estaAdivinada a True e incrementara el valor de cantidadDeAciertos en 1 (uno). De lo contrario restara cantidadDeIntentosPorPalabra.

Formulario

21. Desarrollar todo lo indicado con comentario //Alumno:

Test Unitarios

22. Darle un nombre claro al proyecto, sus clases y sus métodos
23. Agregar 2 test unitarios:
    a. Forzar, mediante el código la ejecución de FileManagerException, validar que suceda de forma correcta.
    b. Al instanciar un nuevo juego, la cantidad de aciertos debe ser igual a 0 (cero).

Al finalizar, colocar la carpeta de la carpeta de la Solución completa en un archivo ZIP que deberá tener como nombre Apellido.Nombre.division.zip y compartir este por Slack sólo con el docente titular de la cursada.