Trabajo Práctico 4 - Clas Guido

Rápido y Furioso - Picadas y Apuestas

Notas previas a ejecutar el programa:
1) Se deberá crear la base de datos "BaseCarreras".
2) Luego se deberá correr el siguiente Query para crear la tabla:

```
USE [BaseCarreras]
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Carreras](
[numCarrera] [int] IDENTITY(1,1) NOT NULL,
[ganador] [varchar](60) NOT NULL,
[corredorElegido] [varchar](60) NOT NULL
) ON [PRIMARY]
GO
```

Descripción del programa:
El programa consta de un sistema de apuestas en base a las carreras entre dos vehículos/corredores, las cuales se denominan picadas. Se comienza con un importe de 100 pesos por defecto, y se debe realizar una apuesta menor a 100 pesos a uno de los dos corredores (Paul Walker y Toretto).

Luego se procede a presionar el botón **Iniciar Picada** el cual iniciara la carrera entre ambos autos, los cuales avanzan en base a su atributo velocidadPunta y un extra de manera random, generado cada vez que se maneja el evento del timer. El primero que llegué al final de la interfaz gráfica, se convertirá en el ganador y se informará si el usuario ganó o perdió su apuesta. Si el usuario perdió todo su dinero, se le informará y se procederá a finalizar el juego.

El botón **Volver a Correr** reiniciará los corredores a los puestos iniciales, y eliminará la apuesta y elección del usuario.

Los botones **Guardar Picadas** y **Reporte de Picadas** se encargan de serializar y guardar en la base de datos un objeto de tipo Carrera, creado a partir del final de la carrera, el cual cuenta con los atributos que indican quien ganó, por quién se apostó y cuánto se apostó. **Reporte de Picadas** brindará al usuario una ayuda sobre a quién conviene apostar en base a los datos serializados.

Por último el botón **Salir**, cerrará el programa automáticamente sin posibilidad de que se guarde alguna acción realizada.

Utilización de temas en el programa:

- Excepciones: Se encuentran a lo largo de todo el programa, específicamente se pueden visualizar en el proyecto que serializa los archivos, y el se encarga de la base de datos. También en el método **Picada**, se utiliza una excepción propia para evitar inconvenientes.
- Test Unitarios: Existe un proyecto de prueba unitaria en la solución, la cual testea dos métodos que se consideraron importantes, como la serialización y conexión a base de datos.
- Generics e Interfaces: Están implementados e incorporados el uno con el otro en un proyecto llamado **Serializadores**, con el fin de facilitar y extender la serialización a distintos tipos de datos. Fue utilizado para serializar Carrera y List<Carrera>.
- Archivos y Serialización: Están anidados en un proyecto llamado **Serializadores**, dentro de la solución. Permite serializar objetos a XML, para luego poder volver a ser leídos e interpretados por el deserializador.
- SQL: El proyecto llamado **SQL**, es el que se encarga de realizar la conexión a la base de datos, y de meramente agregar una lista de carreras, o bien una carrera en particular a la misma.
- Hilos: Se utilizaron los Threads mediante un atributo de tipo List<Thread> que permite almacenar dos hilos en el formulario, uno por cada auto/carril. La idea recae en que cada thread represente un auto, y permita el avance de cada uno en un proceso distinto.
- Eventos y Delegados: Se encuentran puntualmente en el formulario, aunque existen declarados en el proyecto **Entidades**, en las clases **Vehiculo** y **Auto**. Se utilizan tanto para el movimiento, como para manejar los distintos eventos, botones, y elecciones del jugador.
- Método de Extensión: El método de extensión implementado se encuentra en el proyecto **SQL**, bajo la clase **ExtensionString**. Como bien indica, se extiende de la clase string, y permite que utilizando el string con el nombre de la base de datos, se genere el connection string completo.

Recuperatorio Trabajo Práctico 4 - Clas Guido

Nota: En base a correcciones solicitadas por el docente Federico Dávila, se procedió a darle otro enfoque más enriquecedor al programa. Se retocó la acción de ver la picada y su código también, dando algunos cambios de lógica, pero permitiendo que funcione de la manera que se pretende que funcione.

Se minimizó y encapsuló el formulario que corre la picada, de modo que solamente se encargue de hacer eso, y cálculos sobre el ganador o usuario.

El nuevo enfoque se basa en un formulario principal que se divide en tres secciones, “Sección Grillas”, “Sección Apuestas”, y “Mi Cuenta”. Esto permite enfocarse más en las apuestas y no tanto en “como se mueven los autos”. Tener reportes y poder acceder a

información de cada carrera. *Además está en constante manejo de archivos e intercambio de datos entre la BBDD y el archivo XML que almacena las picadas.*

También se pretendió realizar un programa que a su vez permita actualizarse y en un futuro realizar la carga de más corredores o más funcionalidades, por lo que se optó por retocar el código donde se consideró conveniente para que esto sea posible.

Sección Grillas: Esta sección permite acceder a un nuevo formulario que desplegará la grilla de picadas realizadas, las cuales cuentan con datos como, quien ganó, cuanto se apostó, etc. También brinda un reporte en cuanto a la diferencia de victorias de un corredor por sobre el otro, permitiendo dar una pequeña ayuda al jugador sobre a quien conviene apostar.
Por último, cuenta con la opción de que en caso de que la base de datos se encuentre desactualizada con respecto al archivo Xml que guarda las carreras, se pueda actualizar mediante un botón.

Sección Apuestas: Esta sección es la cual se considera más importante, se implementaron validaciones correspondientes que permitan que el jugador no realice acciones no esperadas, o bien se mitigaron los riesgos de estas acciones. Se permite elegir el corredor desde un ComboBox, y el dinero a apostar mediante un NumericUpDown completamente validado entre 0 y el dinero del usuario.
El botón Observar será el nuevo encargado de desplegar un nuevo formulario permitiendo observar la picada, con la única opción de comenzarla, y finalizarla (cuando corresponda).

Sección Mi Cuenta: Está sección es breve y lo que permite es saber al usuario con cuanto dinero cuenta actualmente, ya sea para realizar una nueva apuesta o para verificar si le conviene retirarse con ese dinero. Para esto, se implementó el botón Retirarse, que permite guardar el dinero del jugador en un archivo de texto, y cerrar el programa satisfactoriamente.

Otros cambios:

- Se reformuló el delegado, el evento, y algunos métodos para que se encarguen de trabajar únicamente con el nombre del corredor que se pretende trazar.
- Se agregó implementó una interfaz nueva “IArchivos” y una clase nueva “Texto” que permiten guardar en un archivo de texto datos en formato string.
- En el proyecto SQL se agregó el método “ListarCarreras” para poder leer de la base de datos la lista de carreras y devolverla en formato List<Carrera>.
- Se colocó a los corredores en un enumerado del form principal, permitiendo así agregar corredores nuevos en nuevas implementaciones, o cambiar los actuales.
- Se contemplaron y mitigaron posibles excepciones al manejo de archivos, formatos, etc.
- Por último y no tan relevante, se le dieron cambios estéticos al formulario para que cuente con una interfaz gráfica más amena al usuario.

IMPORTANTE: El nuevo query para poder generar la BBDD y la tabla Carreras es el siguiente.

```
USE [BaseCarreras]
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Carreras](
[numCarrera] [int] IDENTITY(1,1) NOT NULL,
[ganador] [varchar](60) NOT NULL,
[corredorElegido] [varchar](60) NOT NULL,
[apuestaJugador] [int] NOT NULL
) ON [PRIMARY]
GO
```