

# Laboratorio 2

## Trabajo Practico 4

Alumno: Octavio Bill Zito

División: 2°E

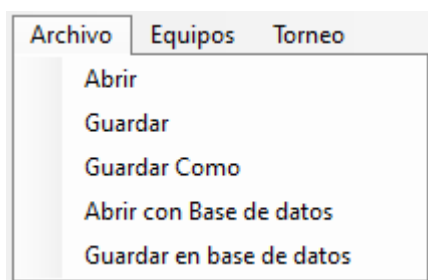
Proyecto: Torneo de Futbol

## Breve resumen:

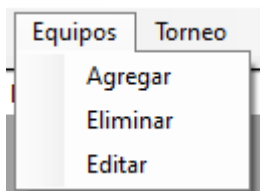
Al iniciar la aplicación nos encontramos con el siguiente form

The screenshot shows a window titled 'Torneo' with a menu bar containing 'Archivo', 'Equipos', and 'Torneo'. Below the menu bar is a table with the following headers: 'Nombre', 'PJ', 'G', 'E', 'P', 'GF', 'GC', 'DG', and 'Pts'. The table body is currently empty. At the bottom of the window, there is status information: 'Fecha: 0/20', 'Torneo: Primera Divison', and 'Capacidad maxima: 15'.

Haciendo click en la parte superior izquierda donde dice archivos podemos abrir, guardar, guardar como, abrir por la base de datos y guardar base de datos.



Al hacer click abrir nos abre una pestaña de Windows para abrir un archivo, que tiene que ser de formato txt, xml o json. Al hacer click en guardar, guardamos el archivo abierto, en caso que no haya un archivo abierto nos pedirá guardar en una ubicación, y guardar como nos abre una pestaña de Windows para guardar en la ubicación que deseamos. Abrir con Base de datos, trae la información de la base de datos y guardar, borra todo lo de la tabla y guarda lo que hay momentáneamente.

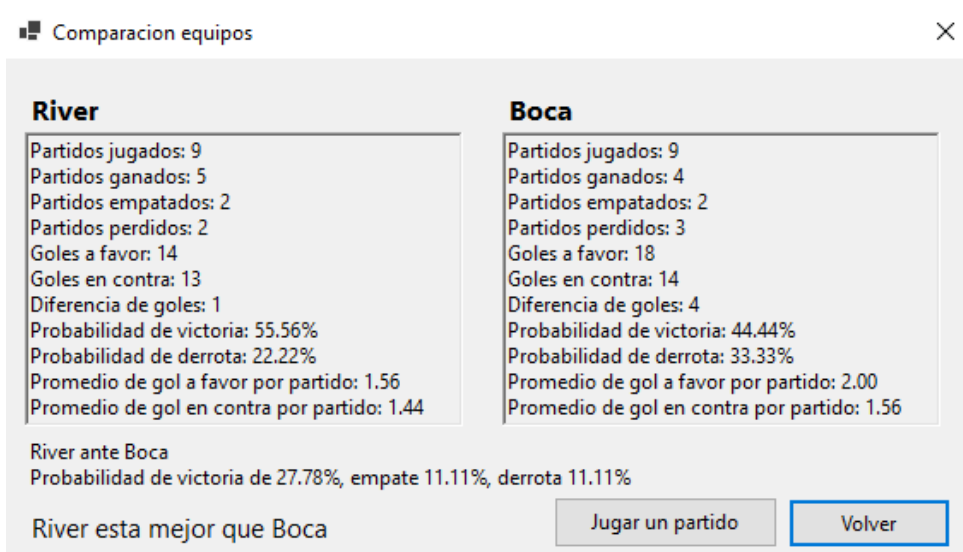


En la pestaña Equipos si hacemos click en agregar nos abre un nuevo form para agregar un equipo, si seleccionamos un equipo y hacemos click en eliminar lo borra y para editar también se necesita selecciona un equipo y abre un form para editar.



En la pestaña Torneo, si hacemos click en avanzar fechas y se encuentra más de 1 equipo y no llegamos a la fecha máxima simula una fecha a los equipos de la tabla.

Al seleccionar dos equipos con el botón Control y tocar en Comparar nos abre un form nuevo con los datos de cada equipo y sus estadísticas.

A screenshot of a software window titled 'Comparacion equipos'. The window displays statistics for two teams: River and Boca. The statistics are presented in two columns, one for each team. At the bottom, there is a summary of the comparison and two buttons: 'Jugar un partido' (Play a game) and 'Volver' (Return).

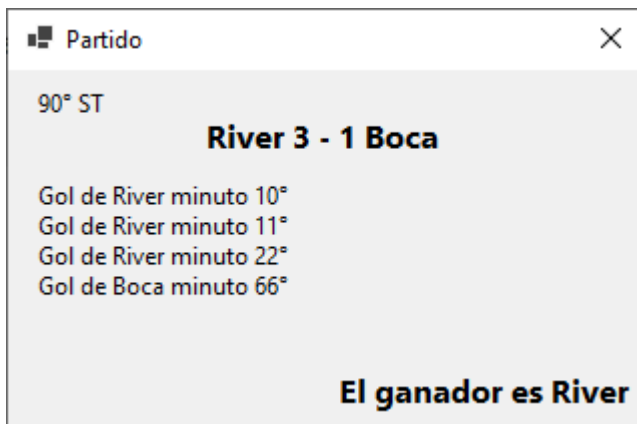
River	Boca
Partidos jugados: 9	Partidos jugados: 9
Partidos ganados: 5	Partidos ganados: 4
Partidos empatados: 2	Partidos empatados: 2
Partidos perdidos: 2	Partidos perdidos: 3
Goles a favor: 14	Goles a favor: 18
Goles en contra: 13	Goles en contra: 14
Diferencia de goles: 1	Diferencia de goles: 4
Probabilidad de victoria: 55.56%	Probabilidad de victoria: 44.44%
Probabilidad de derrota: 22.22%	Probabilidad de derrota: 33.33%
Promedio de gol a favor por partido: 1.56	Promedio de gol a favor por partido: 2.00
Promedio de gol en contra por partido: 1.44	Promedio de gol en contra por partido: 1.56

River ante Boca  
Probabilidad de victoria de 27.78%, empate 11.11%, derrota 11.11%

River esta mejor que Boca

Jugar un partido Volver

Al hacer click en “Jugar Partido”, nos abre una pestaña que simula un partido, teniendo más chance de ganarlo el equipo que tenía más probabilidad de victoria contra el otro.



Resetear Torneo coloca todas las estadísticas del equipo en 0 y vuelve a la fecha 0

El botón info torneo abre un form para cambiar el nombre del torneo, la fecha máxima y la cantidad de equipos

A screenshot of a software window titled "Editar Torneo" with a close button (X) in the top right corner. The window has a light gray background. It contains several input fields: "Nombre del torneo" with the value "Primera Divison", "Cantidad de fechas" with the value "20", and "Cantidad maxima de equipos" with the value "15". Below these fields, it says "Fecha actual: 0". At the bottom, there are two buttons: "Editar" and "Volver".

Y el botón avanzar hasta nos permite simular varias fechas, siempre y cuando no estemos en la fecha final o no haya más de 1 equipo

Avanzar Fechas

Fecha a llegar: 10

Confirmar

Fecha actual: 4

Fecha a llegar: 10

Detener

Cerrar

Temas utilizados Parte 1:

### Excepciones:

ExceptionXml

Class

→ Exception

Fields

clase : string

metodo : string

Properties

Clase { get; } : string

Metodo { get; } : string

Methods

ExceptionXml(string mensaje, string metodo, string clase)

ExceptionXml(string mensaje, string metodo, string clase, Exception innerException)

**ExceptionTxt**  
Class  
↳ Exception

Fields

clase : string

metodo : string

Properties

Clase { get; } : string

Metodo { get; } : string

Methods

ExceptionTxt(string mensaje, string metodo, string clase)

ExceptionTxt(string mensaje, string metodo, string clase, Exception innerException)

**ExceptionJson**  
Class  
↳ Exception

Fields

clase : string

metodo : string

Properties

Clase { get; } : string

Metodo { get; } : string

Methods

ExceptionJson(string mensaje, string metodo, string clase)

ExceptionJson(string mensaje, string metodo, string clase, Exception innerException)

**ExceptionSQL**  
Class  
↳ Exception

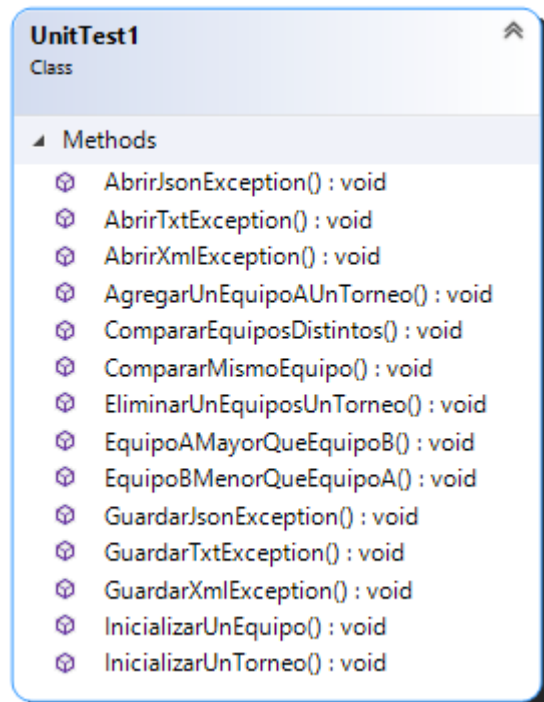
Methods

ExceptionSQL(string message)

ExceptionSQL(string message, Exception innerException)

Las exception de Json, Xml y Txt son utilizadas en la clase GestorDeArchivos, para capturar la excepcion lanzadas por el StreamReader, StreamWriter, XmlTextReader, XmlSerializer y JsonSerializer y relanzarlas con las excepciones propias con un mensaje para capturarla luego en los formularios.

### **Pruebas unitarias:**

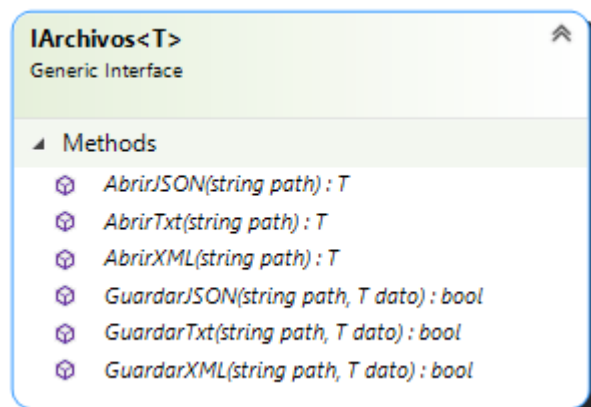


La clase UnitTest es de un proyecto de UnitTest en el cual posee varios métodos para comprobar el funcionamiento de las clases.

### **Tipos genéricos:**

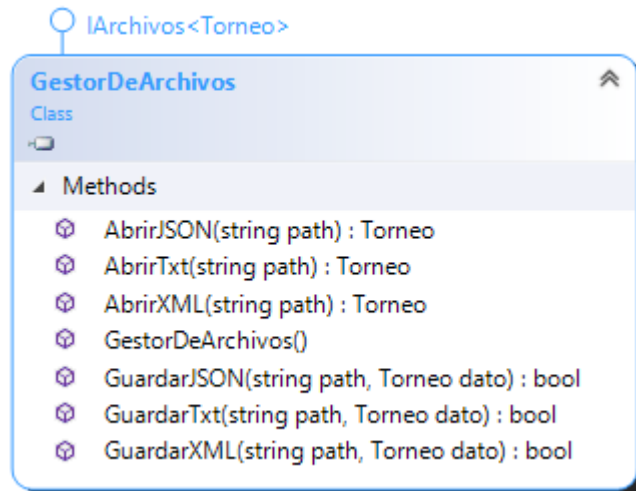
Usada en la interface IArchivos y aplicando su implementación en la clase GestorDeArchivos.

### **Interfaces:**



La interface IArchivos posee un genérico T donde es obligatorio que sea de clase Torneo, posee 6 métodos para ser implementados en el GestorDeArchivos.

### **Archivos:**



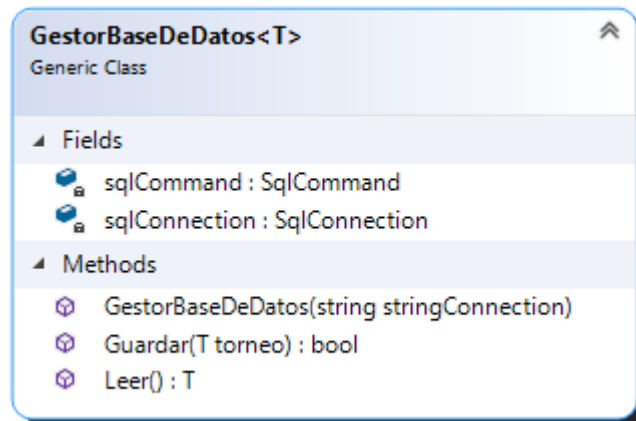
La clase `GestorDeArchivos` implementa la interfaz, esta clase abre y guarda archivos en formato Json, Xml y Txt, en caso de capturar alguna excepción esta es relanzada con las excepciones creadas.

### **Temas utilizados parte 2:**

#### **Base de datos:**

Cree una base de datos con el nombre equipos y una tabla llamada `tablaEquipos`, dejen el script en la carpeta principal del Trabajo Practico 4. La connection string se encuentra en el constructor del `FormPrincipal` al momento de instanciar la clase `GestorBaseDeDatos<Torneo>` con el genérico `Torneo`.

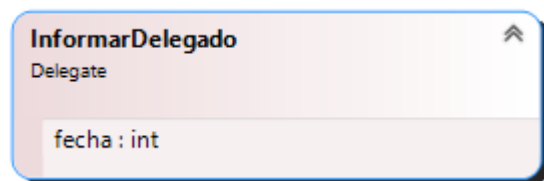




```
this.gdb = new GestorBaseDeDatos<Torneo>("Data Source=.;Initial Catalog=equipos;Integrated Security=true");
```

Línea 31 de la clase FormPrincipal.

### Delegados:



Creo un delegado que retorna void y recibe un int con el nombre fecha. Es utilizado en la clase frmAvanzarFechas en el método Informar. Para rellamar a la función informar.

```
public void informar(int fecha)
{
    if (this.InvokeRequired)
    {
        InformarDelegado del = new InformarDelegado(this.informar);
        object[] args = new object[] { fecha };
        this.Invoke(del, args);
    }
    else
    {
        this.lblFechaActual.Text = "Fecha actual: " + this.torneo.FechaActual.ToString();
        this.progressBar1.Value = fecha;
        this.frmPrincipal.ActualizarForm();
    }
}
```

También el evento, eventoInformar de Torneo es de tipo InformarDelegado.

## Hilos:

En el frmAvanzarFechas si se cumplen ciertas condiciones crea un hilo que ejecute el método AvanzarHasta de la clase torneo pasándole por parámetro la fecha hasta que deseo llegar y un token para poder cancelarlo, este token es creado en el constructor de frmAvanzarFechas.

```
1 reference
private void AvanzarFechas()
{
    if (this.fechaLlegar > this.torneo.FechaActual && this.fechaLlegar <= this.torneo.CantidadDeFechas)
    {
        Task task = new Task(() => this.torneo.AvanzarHasta(this.fechaLlegar, tokenSource.Token));
        task.Start();
    }
}
```

Dentro del método AvanzarHasta se llama al método AvanzarTorneo y si retorna true pasamos a llamar al evento "eventoInformar" pasándole la fecha actual, luego se realiza un Thread.Sleep de 500 milisegundos y si la FechaActual no es igual a la fechaAvanzar y el token no fue cancelado sigue el proceso.

```
1 reference
public bool AvanzarHasta(int fechaAvanzar, CancellationToken token)
{
    do
    {
        if(this.AvanzarTorneo())
        {
            if (eventoInformar != null)
            {
                this.eventoInformar.Invoke(this.FechaActual);
            }
        }else
        {
            return false;
        }
        Thread.Sleep(500);
    } while (fechaAvanzar != this.FechaActual && !token.IsCancellationRequested);

    return true;
}
```

## Eventos:

En la clase Torneo creo un evento del tipo InformarDelegado como se mostró previamente

```
public event InformarDelegado eventoInformar;
```

En el cual es llamado dentro del método AvanzarHasta de la misma clase, su implementación se encuentra en el frmAvanzarFechas.

```
this.torneo.eventoInformar += this.informar;
```

Suscribiéndole el método informar del formulario

```
public void informar(int fecha)
{
    if (this.InvokeRequired)
    {
        InformarDelegado del = new InformarDelegado(this.informar);
        object[] args = new object[] { fecha };
        this.Invoke(del, args);
    }
    else
    {
        this.lblFechaActual.Text = "Fecha actual: " + this.torneo.FechaActual.ToString();
        this.progressBar1.Value = fecha;
        this.frmPrincipal.ActualizarForm();
    }
}
```

Este primero necesita ser invocado al hilo principal, luego de eso modifica el valor de la barra de progreso, actualiza el datagrid y cambia el valor del label.

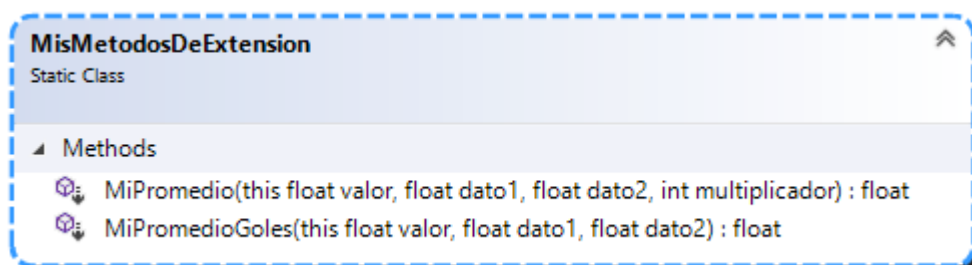
En el FormClosing de frmAvanzarFechas, si el hilo se encuentra ejecutándose se detiene y se le quite el manejador del evento

```
private void frmAvanzarFechas_FormClosing(object sender, FormClosingEventArgs e)
{
    this.tokenSource.Cancel();
    this.torneo.eventoInformar -= this.informar;
}
```

También al momento de hacer click en el botón detener de este mismo form se llama al método Cancel del tokenSource.

### **Métodos de extensión:**

En la clase MisMetodosDeExtension que es static poseemos dos métodos para sacar promedios estos métodos son utilizados en FormComparar para sacar los promedios



```
probGanados = probGanados.MiPromedio((float)this.e1.Ganados, (float)this.e1.PartidosJugados, 100);
```

```
promGolesFavor = promGolesFavor.MiPromedioGoles((float)this.e1.GolesAFavor, (float)this.e1.PartidosJugados);
```