Primera entrega trabajo final
Seminario .NET

Blanco Valentín (24549/6)
Cabe Pablo (23056/4)
Isla Federico (26241/6)
Manese Martín (23105/5)

Mediante este texto se detallará la funcionalidad del trabajo desarrollado desde Program.cs.

Aclaraciones

Los archivos de texto referidos a la persistencia de datos y los archivos de texto de IDs se crearán en la ruta **CentroEventos.Consola\bin\Debug\net8.0**

Se decidió utilizar archivos de texto para persistir los IDs. Además, se aplicó una clase de utilidad para el manejo de ellos (**IdManager.cs**). Esta misma emplea dos métodos:

- BuscarUltimoId: si el archivo no existe retorna -1. En caso contrario devuelve el último id.
- ActualizarArchivoId: actualiza el último archivo o lanza la excepción si el archivo no existe o no es posible escribirlo.

Cada repositorio adquirirá la responsabilidad de autoincrementar los respectivos IDs.

Funcionalidad

En primer lugar, se instanciarán los **repositorios** ubicados en CentroEventos.Repositorios y los **casos de uso**, situados en CentroEventos.Aplicacion.CasosDeUso.

Cabe destacar que los repositorios utilizan las **interfaces** creadas en el directorio CentroEventos.Aplicacion.Interfaces, con la intención de aplicar la inyección de dependencias. Esta resultará útil de cara al futuro para la implementación de otro servicio de persistencia.

```
using CentroEventos.Aplicaciones.Excepciones;

IRepositorioPersona repoPersona= new RepositorioPersona();
IServicioAutorizacion servicioAutorizacion = new ServicioAutorizacionProvisorio();
IRepositorioEventoDeportivo repoEvento = new RepositorioEventoDeportivo();
IRepositorioReserva repoReserva = new RepositorioReserva();

AltaPersonaUseCase altaPersona = new AltaPersonaUseCase(repoPersona, servicioAutorizacion);
ModificarPersonaUseCase modPersona = new ModificarPersonaUseCase(repoPersona, servicioAutorizacion);
BajaPersonaUseCase bajaPersona = new BajaPersonaUseCase(repoPersona,repoEvento,repoReserva, servicioAutorizacion);

AltaEventoUseCase altaEvento = new AltaEventoUseCase(repoEvento, repoPersona, servicioAutorizacion);
ModificarEventoUseCase modEvento = new ModificarEventoUseCase(repoEvento,repoPersona,servicioAutorizacion);
BajaEventoUseCase bajaEvento = new BajaEventoUseCase(repoEvento, repoPersona, servicioAutorizacion);

AltaReservaUseCase altaReserva = new AltaReservaUseCase(repoReserva, repoPersona, repoEvento, servicioAutorizacion);
ModificarReservaUseCase modReserva = new ModificarReservaUseCase(repoReserva, repoPersona, repoEvento, servicioAutorizacion);
BajaReservaUseCase bajaReserva = new BajaReservaUseCase(repoReserva, repoPersona, repoEvento, servicioAutorizacion);

ListarPersonasUseCase listarPersonas = new ListarPersonasUseCase(repoPersona);
ListarEventosUseCase listarEventos = new ListarEventosUseCase(repoEvento);
ListarReservaUseCase listarReservas = new ListarReservaUseCase(repoReserva);

ListarAsistenciaAEventoUseCase listarAsistencia = new ListarAsistenciaAEventoUseCase(repoEvento, repoReserva, repoPersona);
ListarEventosConCupoDisponibleUseCase listarEventosConCupo = new ListarEventosConCupoDisponibleUseCase(repoEvento, repoReserva);
```

Una vez instanciados los casos de uso y los repositorios, se emplea un try catch para probar la persistencia de datos de las entidades correspondientes. En este paso, se cargan las personas, los eventos y las reservas.

Código de ejemplo

```
CargarPersonas();
modPersona.Ejecutar(new Persona("9876543", "Juan", "Perez", "juan.perez@gmail.com", "01122334455"), 1);
bajaPersona.Ejecutar(5, 1);
ListadoPersona();
```

```
void CargarPersonas()
{
    altaPersona.Ejecutar(new Persona("9876543", "Juan", "Perez", "juan.perez@example.com", "01122334455"), 1);
    altaPersona.Ejecutar(new Persona("8765432", "Maria", "Gomez", "maria.gomez@example.com", "02299887766"), 1);
    altaPersona.Ejecutar(new Persona("7654321", "Carlos", "Rodriguez", "carlos.rodriguez@example.com", "03311223344"), 1);
    altaPersona.Ejecutar(new Persona("6543210", "Laura", "Fernandez", "laura.fernandez@example.com", "04455667788"), 1);
    altaPersona.Ejecutar(new Persona("5432109", "Sofia", "Martinez", "sofia.martinez@example.com", "05566778899"), 1);
}
```

```
void ListadoPersona()
{
    Console.WriteLine("-----------Personas-----------" + "\n");
    List<Persona> personas = listarPersonas.Ejecutar();
    if (personas != null)
    {
        foreach (Persona p in personas)
        {
            Console.WriteLine(p.ToString().Replace("#", " "));
        }
    }
    else
    {
        Console.WriteLine("No hay personas cargadas");
    }

}
```

En primer lugar, se cargarán las personas mediante el caso de uso AltaPersona.Ejecutar. Luego se modifica la persona y por último se da de baja la persona N°5 (Sofía), por lo cual se imprimirá el listado sin ella, tal como se observa a continuación:

```
----------Personas----------

1 9876543 Juan Perez juan.perez@gmail.com 01122334455
2 8765432 Maria Gomez maria.gomez@example.com 02299887766
3 7654321 Carlos Rodriguez carlos.rodriguez@example.com 03311223344
4 6543210 Laura Fernandez laura.fernandez@example.com 04455667788
```

En segundo paso, cargamos los eventos a través del caso de uso AltaEvento.Ejecutar. Cabe destacar que la entidad EventoDeportivo cuenta con la propiedad FechaHoraInicio, la cual al asignarla es necesario agregar dos segundos a la fecha actual para que sea posible listar las personas de los eventos vencidos.

Aclaración: es necesario ejecutar el Program.cs y esperar dos segundos para volver a ejecutarlo debido a la validación VerFecha, la cual determina si la fecha a crear/modificar es mayor o igual a la fecha actual. En otras palabras, es imposible ejecutarlo sin esperar el tiempo vencido. Asimismo, para ejecutar el método se deben comentar los métodos que modifiquen a los archivos.

```
CargarEventos();
modEvento.Ejecutar(new EventoDeportivo("Futbol", "Final de la copa interfacultades", DateTime.Now.AddSeconds(2), 10, 22, 1), 1);
bajaEvento.Ejecutar(repoReserva, 4, 1);
ListadoEvento();
```

```
void CargarEventos()
{
    altaEvento.Ejecutar(new EventoDeportivo("Futbol", "Final copa interfacultades", new DateTime(2025, 8, 25, 12, 0, 0), 6,3, 1), 1);
    altaEvento.Ejecutar(new EventoDeportivo("Voley", "UNLP vs UBA", new DateTime(2025, 11, 7, 21, 0, 0), 3, 2, 4), 1);
    altaEvento.Ejecutar(new EventoDeportivo("Ping Pong", "Demostracion a beneficio", new DateTime(2025, 6, 29, 16, 0, 0), 6, 2, 2), 1);
    altaEvento.Ejecutar(new EventoDeportivo("Tiro con arco", "Fecha 1", new DateTime(2025, 10, 1, 8, 0, 0), 4, 4, 3), 1);

}
```

```
----------Eventos----------

1 Futbol Final de la copa interfacultades 20/05/2025 07:36:39 p. m. 10 22 1
2 Voley UNLP vs UBA 07/11/2025 09:00:00 p. m. 3 2 4
3 Ping Pong Demostracion a beneficio 29/06/2025 04:00:00 p. m. 6 2 2
```

En tercer lugar, con respecto a las reservas, el funcionamiento es similar al de cargarPersona. A continuación se realiza la baja de la reserva con ID 4.

```
CargarReservas();
modReserva.Ejecutar(new Reserva(1, 1, DateTime.Now, EstadosAsistencia.Presente), 1);
bajaReserva.Ejecutar(4, 1);
ListadoReservas();
```

```
void CargarReservas()
{
    altaReserva.Ejecutar(new Reserva(1, 1, DateTime.Now, EstadosAsistencia.Pendiente),1);
    altaReserva.Ejecutar(new Reserva(2, 1, DateTime.Now, EstadosAsistencia.Pendiente),1);
    altaReserva.Ejecutar(new Reserva(3, 1, DateTime.Now, EstadosAsistencia.Pendiente),1);
    altaReserva.Ejecutar(new Reserva(1, 2, DateTime.Now, EstadosAsistencia.Pendiente),1);

}
```

```
----------Reservas----------

1 1 1 20/05/2025 07:36:38 p. m. Presente
2 2 1 20/05/2025 07:36:38 p. m. Pendiente
3 3 1 20/05/2025 07:36:38 p. m. Pendiente
```

ListadoEventosDisponibles: en este se implementa el caso de uso. ListarEventosConCupoDisponible, el cual muestra los eventos que poseen disponibilidad de cupo.

ListarPersonasAsistidas: se muestran las personas presentes en el evento con ID = 1. En este sentido, es importante mencionar que para probar este último método es necesario comentar (//) todos los otros.

```
ListadoEventosDisponibles();
ListarPersonasAsistidas(1);
```

Salida esperada por consola

La primera vez que se ejecuta se verá de la siguiente manera:

```
----------Personas----------

1 9876543 Juan Perez juan.perez@gmail.com 01122334455
2 8765432 Maria Gomez maria.gomez@example.com 02299887766
3 7654321 Carlos Rodriguez carlos.rodriguez@example.com 03311223344
4 6543210 Laura Fernandez laura.fernandez@example.com 04455667788
----------Eventos----------

1 Futbol Final de la copa interfacultades 21/05/2025 10:17:27 p. m. 10 22 1
2 Voley UNLP vs UBA 07/11/2025 09:00:00 p. m. 3 2 4
3 Ping Pong Demostracion a beneficio 29/06/2025 04:00:00 p. m. 6 2 2
----------Reservas----------

1 1 1 21/05/2025 10:17:26 p. m. Presente
2 2 1 21/05/2025 10:17:26 p. m. Pendiente
3 3 1 21/05/2025 10:17:26 p. m. Pendiente
----------Eventos con cupo disponibles----------

1 Futbol Final de la copa interfacultades 21/05/2025 10:17:27 p. m. 10 22 1
2 Voley UNLP vs UBA 07/11/2025 09:00:00 p. m. 3 2 4
3 Ping Pong Demostracion a beneficio 29/06/2025 04:00:00 p. m. 6 2 2
----------Personas asistidas al evento 1----------

El evento todavía no ocurrió.
```

La segunda vez es necesario comentar todos los métodos salvo el último (ListarPersonasAsistidas), para verlo de la siguiente forma:

```
try
{
    //CargarPersonas();
    //modPersona.Ejecutar(new Persona("9876543", "Juan", "Perez", "juan.perez@gmail.com", "01122334455"), 1);
    //bajaPersona.Ejecutar(5, 1);
    //ListadoPersona();
    //CargarEventos();
    //modEvento.Ejecutar(new EventoDeportivo("Futbol", "Final de la copa interfacultades", DateTime.Now.AddSeconds(1), 10, 22, 1), 1);
    //bajaEvento.Ejecutar(repoReserva, 4, 1);
    //ListadoEvento();
    //CargarReservas();
    //modReserva.Ejecutar(new Reserva(1, 1, DateTime.Now, EstadosAsistencia.Presente), 1);
    //bajaReserva.Ejecutar(4, 1);
    //ListadoReservas();
    //ListadoEventosDisponibles();
    ListarPersonasAsistidas(1);


}
catch (Exception e)
{
  Console.WriteLine(e.Message);
}
```

```
------------Personas asistidas al evento 1-----------

1 9876543 Juan Perez juan.perez@gmail.com 01122334455
```

Esta es la salida del método que se debe ejecutar pasados dos segundos.