TP PARADIGMA ORIENTADO A OBJETOS

UADE

Profesor: Sauczuk, Martin Pablo

Integrantes:

- Berns, Erik Christian – LU: 1173293
- Liñeira Abal, Tomas – LU: 1164225
- Lussoro, Nicolás – LU: 1169472
- Sanchez, Ivan Lautaro – LU: 1164700
- Valentini, Tiago – LU: 1167213

Fecha de entrega: 16 de noviembre de 2024

Índice

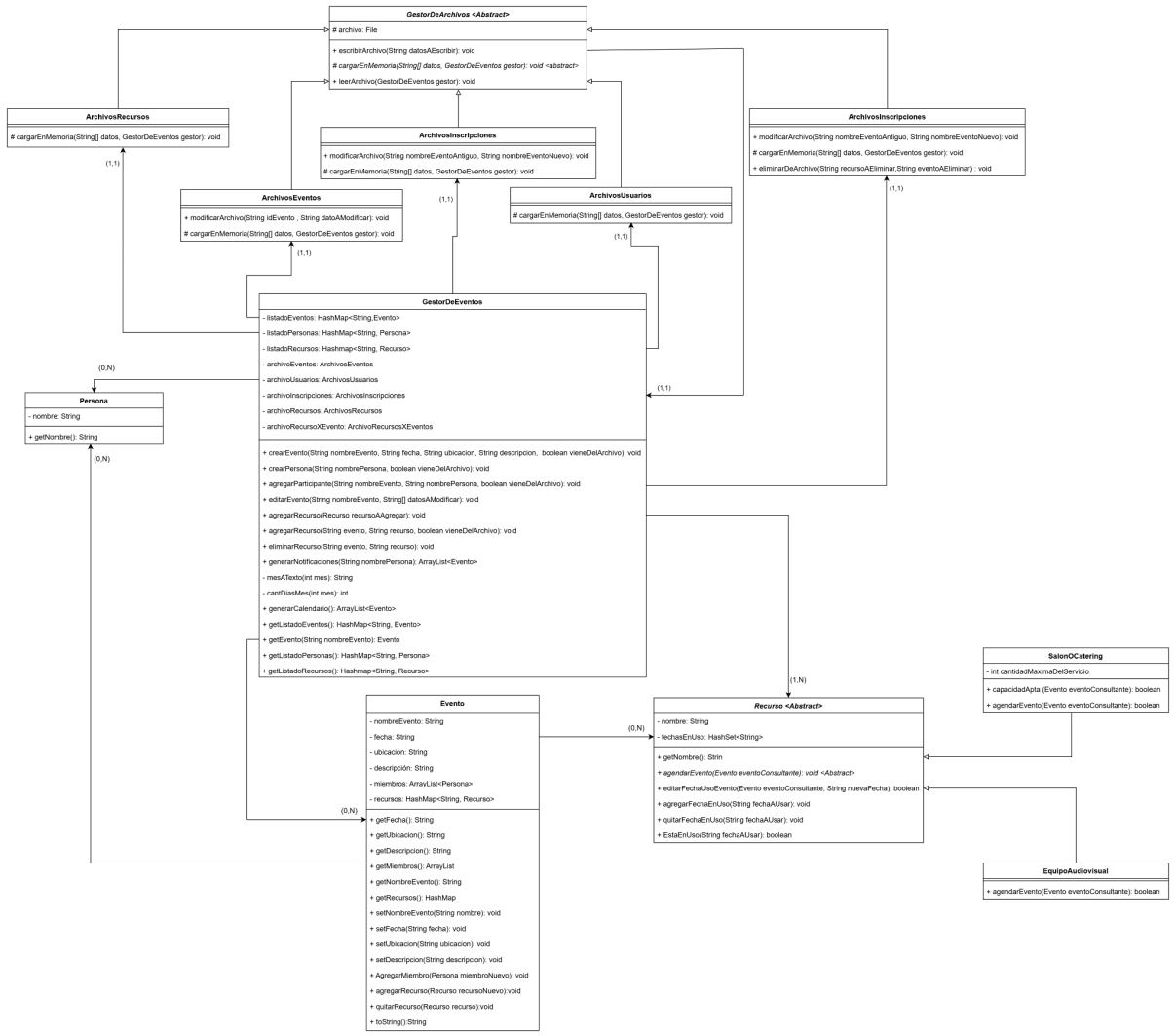
Introducción

En el siguiente trabajo se va a resolver el problema proporcionado a la cátedra el cual es realizar un gestor de eventos capaz de administrar los mismos, usuarios y recursos.

Dentro del informe se van a explicar las distintas estructuras que lo componen y su funcionamiento. Además, se proporcionará un enlace a un repositorio con la estrategia en el lenguaje de la materia (Java)

Desarrollo de la estrategia

Diagramas de clases



Por motivo de lectura, el diagrama también está disponible en el repositorio del proyecto.

Clases

GestorDeEventos

El gestor de eventos es la clase más importante ya que se encarga de gestionar todos las clases de la aplicación (Eventos, Personas, Recursos y Archivos), permitiendo generar eventos, personas o incluso un calendario. También permite agregar participantes y recursos a un evento, pudiendo así editar el mismo.

Evento

Esta clase contiene toda la información perteneciente a un evento como el nombre, fecha, ubicación y descripción del evento, así como los miembros participantes del evento y los recursos destinados al mismo. Además de sus respectivos getters y setters, esta clase permite agregar miembros y agregar o quitar recursos del evento.

Persona

Esta clase representará, como su nombre lo dice, a una persona dentro del sistema y tiene como atributo un string “nombre”.

Recurso

Esta clase abstracta representa un recurso con restricciones de uso por fecha, asegurando la unicidad de fechas ocupadas. Utiliza el atributo “nombre” para identificar el recurso, y métodos para gestionar fechas, ya sea para agregar, verificar o quitar en base a un evento, y además, permite editar las fechas ya reservadas. Asimismo incluye el método abstracto “agendarEvento”, que permite a cada tipo de recurso aceptar un evento en función de sus necesidades (Para el caso un salón o catering aceptaran en función de los invitados y la fecha mientras que los equipos audiovisuales en función de la fecha únicamente).

GestorDeArchivo

Clase abstracta que actúa como padre de todos los archivos del sistema (ArchivosEventos, ArchivosPersona, ArchivosRecurso, ArchivosIncripciones y ArchivosRecursoXEvento). La misma cumple el rol de asignarle a sus hijos la habilidad de leerse y escribirse. Sin embargo,

deja en manos de cada uno como carga sus datos en memoria. Además, dependiendo el tipo de archivo, tendrá el comportamiento de modificar o eliminar datos de su base.

Funcionalidades de la estrategia

El sistema planteado en la solución será capaz de realizar las siguientes funcionalidades:

1. Crear un nuevo evento
2. Modificar un evento
3. Ver eventos
4. Ver registro de personas de cierto evento
5. Inscribir una persona a un evento
6. Gestionar recursos
7. Ver calendario
8. Notificaciones
9. Agregar persona del sistema

A continuación procederemos a explicar cada una y su estrategia

Crear evento

Crea una nueva instancia Evento con sus respectivos datos. Esto lo hace de la siguiente manera:

- Se le consulta al usuario que ingrese los datos del evento a través de un formulario
- El gestor recibe los datos del evento
- Lo guarda en el listado de eventos (el cual es atributo del gestor de eventos)
- Se cuestiona si viene del archivo (a través de un atributo) con el fin de guardar los datos nuevos y no repetir tuplas
- Por último, de no venir del archivo, se lo agrega al archivo correspondiente (en este caso “archivoEventos”).

Modificar evento

Esta función pide el nombre del evento, verifica si existe en el sistema, si existe se llama a la función editarEvento del gestor de eventos, el cual tiene como parámetros, el nombre del evento y una función formulario la cual devuelve el array con los datos a modificar (si no se desea modificar cierto campo se debe ingresa “X”).

Dentro del gestor de eventos, esta función obtiene el evento a modificar y luego verifica las posiciones del array preguntando si es una “X” o un string a modificar:

- Si se quiere modificar el nombre, se cambia el nombre del objeto en sí y elimina, del gestor, el nombre anterior y agrega el nuevo como clave. Luego se actualiza en el archivo de las inscripciones y recursosXeventos el nombre del mismo.
- Si se quiere modificar la fecha, también se modificará la fecha de uso de sus recursos, si están en uso en la nueva fecha los va a liberar
- Los demas parametros simplemente se cambian

Al final se actualiza el evento modificado en la persistencia de datos.

Ver eventos

Esta funcionalidad muestra todos los eventos habidos en el gestor hasta el momento. Esto lo logra de la siguiente manera:

- El gestor a través de un getter le otorga al usuario los eventos del sistema en formato de HashMap.
- El usuario muestra por pantalla con un iterador todos los eventos de la colección

Ver registro de personas de cierto evento

Esta función revisará de que el evento seleccionado se encuentre en el sistema, y, en caso de estarlo, se recorrerá el listado de participantes del evento especificado mediante un ‘Iterator’ y se imprimirán todos sus participantes.

Inscribir una persona a un evento

Esta función se asegura de que el evento al que se desea inscribir a la persona exista y, de ser así, mientras que el usuario no ingrese la letra “X”, se le pedirá al gestorDeEventos que agregue las personas al evento, siempre y cuando estas estén en el sistema.

El gestorDeEventos, cuando le hagan la petición, agrega la persona solicitada a la colección de personas del evento pedido; para posteriormente fijarse que los recursos que posee no entren en conflicto con la petición. De entrar en conflicto lo avisará por pantalla y procederá a quitarle el recurso.

Gestionar recursos

Esta función se encarga de agregar y quitar recursos a un evento, el cual se indica mediante una entrada por teclado. Luego se solicita que ingrese la letra A para agregar un recurso, Q para quitar un recurso o cualquier otra tecla para salir.

En caso de haber elegido agregar un recurso se muestran todos los recursos y se solicita que ingrese aquel recurso que desea agregar a el evento.

Por otro lado, si elige quitar un recurso del evento tan solo se mostraran como opciones los recursos que ya tiene asignado ese evento y se solicita que ingrese aquel recurso que desea quitar del evento

Ver calendario

Mediante una función llamada generarCalendario, en la cual toma la fecha actual y selecciona los eventos que se encuentran en ese mes y ese año. Luego, ordena mediante el comparable implementado en eventos, el cual ordena por fechas, para luego imprimir de forma gráfica el calendario.

Finalmente, utiliza la función listarEventos en la cual se muestran por pantalla los detalles de todos los eventos seleccionados anteriormente.

Notificaciones

Esta función solicita al usuario ingresar el nombre de una persona registrada, verificando que esté en sistema, para luego generar una lista de eventos relacionados con esa persona. Mediante la función listarEventos mostrarlos por pantalla.

Para generar esta lista el gestor de eventos toma los eventos en sistema y la fecha actual para, ir agregando aquellos eventos posteriores a la fecha actual donde está anotada la persona solicitada. Para finalizar ordena los eventos mediante un comparable incorporado a ese objeto, para comparar mediante fecha del mismo.

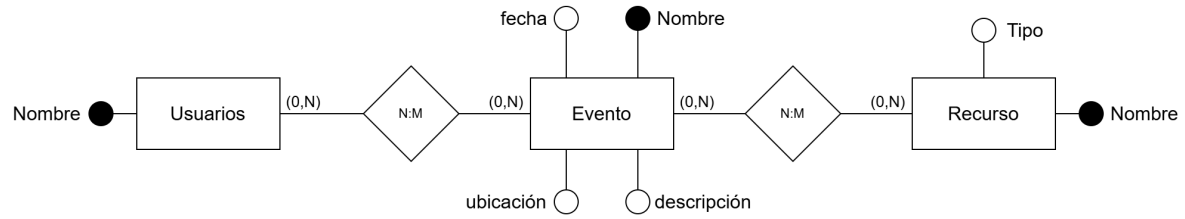
Agregar persona del sistema

Agrega una persona al sistema primero asegurando que el nombre sea único, emplea un Map para verificarlo y pide un nombre no ingresado anteriormente en caso de que ya se haya ingresado anteriormente. Cuando se ingresa un nombre válido, se llama a "crearPersona" para registrar a la persona.

Material extra a la materia investigado

Persistencia de datos

Nosotros como grupo decidimos tomar el desafío de que los datos creados por el programa pudieran persistir en archivo de texto. Por lo que primero nos pusimos a pensar cómo sería esta "base de datos" con la que interactuar el sistema y llegamos al siguiente Diagrama Entidad Relación (DER).



Para implementar en java, pensamos a cada tabla del modelo relacional como un objeto. Sin embargo notamos que había funciones iguales en estrategia y funciones que no debido a la cantidad de columnas de la tabla. Ante esto nos vimos en la necesidad de aplicar una clase abstracta capaz de agrupar estas funciones en común y dejar a las distintas tablas hacer sus comportamientos particulares.

Las acciones que hacen en común todas las tablas es leerse, escribirse y cargar en memoria los datos (en función del hijo). Mientras que dependiendo la tabla implementa las funciones de modificar o eliminar elementos.

Para realizar estas acciones los objetos se ven ligados a usar herramientas de java para el manejo de archivos. Estas son la clase File para manejar archivos, FileReader y BufferedReader para leerlos y FileWriter para escribir en ellos.

Entradas de datos a sistema

Para implementar la entrada de datos por teclado al sistema nos decidimos por utilizar la clase Scanner que ya viene incluida en la instalación de Java, la cual tras ser instanciada permite el ingreso de datos por teclado por medio de la terminal mediante el método nextLine() el cual recoge los datos insertados permitiendo su posterior manipulación.

Formato de texto

Para que los resultados por pantalla se vieran atractivos y simples de visualizar investigamos sobre cómo editar texto en la consola de Java y encontramos dos elementos que nos interesaron.

En primer lugar encontramos el método String.format(). el cual nos permite renderizar strings. El mismo pedirá la plantilla del String a renderizar dejando los espacios a completar con el carácter %[espacio en el que se inserta][tipo de dato] para después pedirlos en los atributos consiguientes. Esto nos permitió darle más armonía a las devoluciones del gestor

Por otra parte, para complementar lo anterior indagamos sobre el código ANSI. Una secuencia de caracteres que interpreta los Strings para cambiar los colores de la consola. Dando un resultado más visual y legible a las respuestas.

Funciones de tiempo

Para implementar ciertas funciones relacionadas con el tiempo actual como por ejemplo el calendario y/o las notificaciones utilizamos la clase LocalDate la cual contiene el método now() el cual devuelve la fecha actual en formato YYYY-MM-DD, facilitando así comparaciones entre fechas para el filtrado de resultados

Enlace de proyecto

https://github.com/1tiagovalentini/TPO-INTEGRAL-POO.git

Conclusiones

Si descartamos la persistencia de datos y tratamos a cada recurso de la misma manera podríamos resumir el sistema a 4 clases siendo estas Persona, Evento, Recurso y GestorDeEvento.

Decidimos no desarrollar esa versión simplificada, siendo que la persistencia de los datos hace que este sistema tenga sentido. Pues de no retener los datos, se vuelve totalmente ineficiente. De igual manera dada la ambigüedad de la consigna (para con la forma en la que hay que gestionar los recursos) decidimos darle algo más de complejidad para que esta pueda tener algo de sentido (añadiendo disponibilidad en función de fechas y capacidad apta del recurso, es decir, no puedo dar a un evento de 200 personas un salón donde solo caben 20), pero en consecuencia la cantidad de clases aumentó a 12.

Por ello decidimos extender el sistema en pos de darle una forma más coherente, asegurándonos igualmente de que las clases adicionales tengan sentido, siguiendo los lineamientos vistos en clase.