**Universidad de Santiago de Chile**

**Facultad de Ingeniería**

**Departamento de Ingeniería Informática**

**Paradigmas de Programación**

Laboratorio 3 Paradigmas de Programación

Autor: Mauricio Valdés

Docente: Daniel Gacitúa

24/01/2022

Tabla de Contenido

1. Introducción 3
2. Descripción del problema 3
3. Descripción del Paradigma 4
4. Análisis del Problema 4
5. Diseño de la solución 6
6. Aspectos de Implementación 7
7. Instrucciones de Uso 7
8. Resultados y Autoevaluación 10
9. Conclusión 10

Introducción

Hoy en día debido a la globalización del internet, la gran mayoría de personas cuentan con acceso a este, por esto se ha masificado el uso de plataformas que ofrecen servicios online, un ejemplo de este es Google, el cual tiene diferentes funciones y servicios, dentro de estos tenemos Google Docs, Google Drive, etc.

Debido a esto se requiere desarrollar una plataforma que imite el funcionamiento de Google Docs, el cual es un editor de texto como Word, pero de manera online, en el cual puedes crear documentos y añadir colaboradores.

Para esto realizaremos el desarrollo de esta plataforma en diferentes lenguajes enfocados en distintos paradigmas y así ver cuál es la manera más efectiva y eficiente, y poder notar en que se especializa cada paradigma y en qué caso podríamos usarlos en nuestra vida laboral.

En el desarrollo de este laboratorio se utilizará java para poder realizar el problema indicado, java es un lenguaje orientado a objetos, este fue desarrollado por james Gosling, este lenguaje data del año 1996.

La programación orientada a objetos se basa en el concepto de crear un modelo del problema de destino en sus programas. La programación orientada a objetos disminuye los errores y promociona la reutilización del código.

Creación de objetos basada en clases. Las clases son plantillas para la creación de objetos. Los objetos son estructuras de datos con el comportamiento asociado.

Descripción del problema

Para este semestre se ha definido hacer un editor de texto colaborativo como Google Docs. Los documentos colaborativos permiten a uno o más usuarios compartir documentos y hacer ediciones sobre este. Dentro de las operaciones que típicamente se pueden realizar en estas plataformas tecnológicas se encuentran:

1. Crear una cuenta
2. Iniciar sesión
3. Crear Documentos
4. Eliminar Documentos
5. Compartir Documentos
6. Asignar permisos sobre un document
7. Editar un documento
   1. Escribir
   2. Cambiar estilos a los textos (ej: cursiva, negrita, etc.)
   3. Buscar y reemplazar
   4. Copiar texto
   5. Pegar texto
   6. Cortar texto
   7. Crear títulos, subtítulos
   8. Crear un índice

El objetivo para este problema será desarrollar algunas de estas funciones dentro del paradigma orientado a objetos (POO).

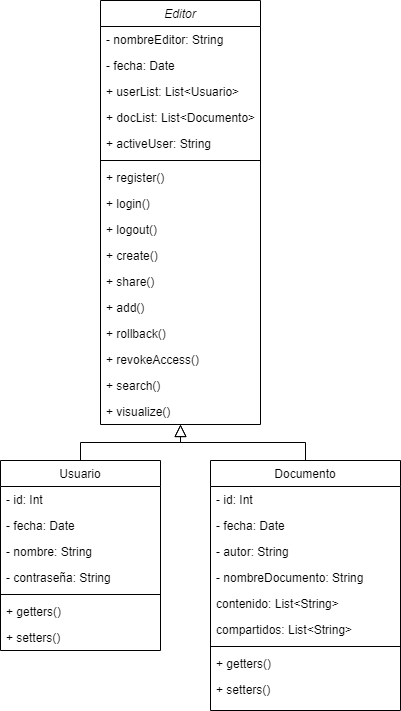
Descripción del paradigma

El paradigma orientado a objetos se basa en la creación de clases y objetos que puedan representar los atributos y métodos de cada una de las abstracciones de la vida real necesarias para poder representar el problema y su solución. Para poder programar en este lenguaje, es necesario analizar el problema para poder obtener las abstracciones de este mismo, analizando los atributos y métodos que puede tener. Luego, se procede a crear clases en base a estas abstracciones. Finalmente, se crean instancias u objetos de estas clases, como representaciones reales (cuantas sean necesarias) para dar con la solución al problema.

Análisis del problema

La problemática del laboratorio es lograr desarrollar las funciones: register, login y logout, estas 3 pertenecen a un apartado que llamaremos authentication, create, share, add, rollback, revokeAccess, search, visualize, etc. Para poder desarrollar estas funciones primero se deben implementar abstracciones mediante TDAs, los cuales serán en base en clases, todo esto dentro del lenguaje java utilizando el paradigma orientado a objetos (POO).

Para implementar los TDAs se deben preguntar ¿Qué contiene un editor de texto como este?, como se mencionó antes los editores de texto de este tipo, de manera online y colaborativos, contienen generalmente un usuario, documentos y los colaboradores de cada documento, por lo tanto, se deberán realizar abstracciones de estos puntos, además se tendrá que implementar un TDA para el editor que es donde se almacenara el nombre tanto del editor de texto como su lista de usuarios y acciones que realicen estos, esta funcionara como un stack en donde se almacenara todo.



Diseño de la solución

Luego de analizar el problema y responder las preguntas básicas se deben implementar las abstracciones, las cuales serían los TDAs, en primer lugar, se presentar un usuario el cual será una clase la cual tendrá como atributos los siguientes: id, fecha, nombre y contraseña.

Luego en segundo lugar se encuentran los documentos, los cuales serán una clase con los siguientes atributos: id, fecha, autor, nombre de documento, lista de contenidos y lista de compartidos.

Como se observa dentro de estos TDAs se encuentra una fecha para la cual se utilizará la clase tipo Date ya definida dentro de java.

Por otro lado, se requiere de un TDA Editor, este TDA se utilizará para la creación del editor colaborativo y contará con los siguientes parámetros: nombre del editor, fecha de creación, lista de usuarios, lista de documentos, y el usuario activo en el momento. Esta último String es el que se utilizara para el desarrollo del resto de funciones, esto debido a que se necesita que el usuario este activo para poder acceder a estas funciones.

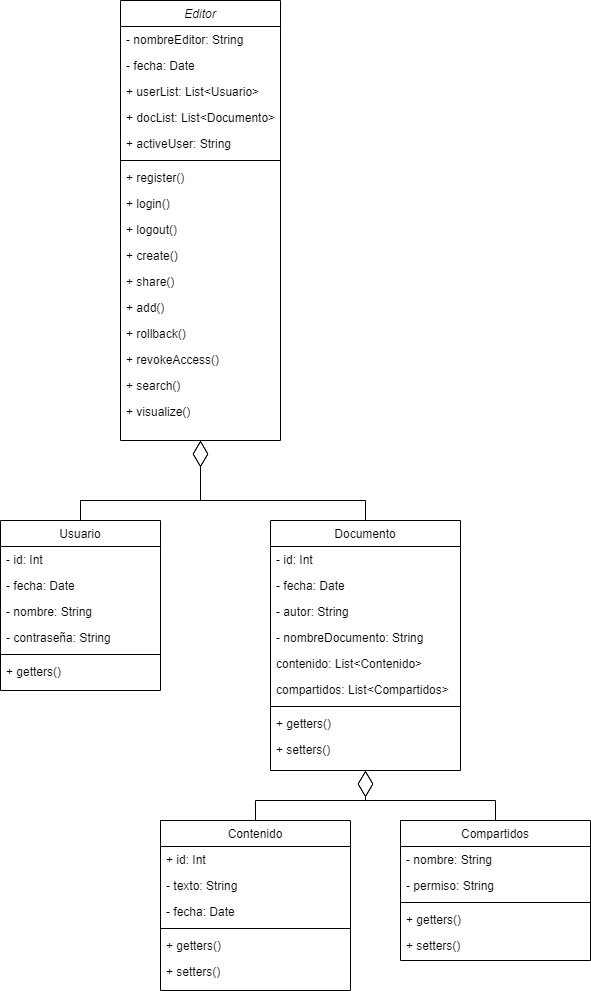
Para el desarrollo de las funciones que tendrá cada usuario dentro de la red social se debe utilizar este editor anteriormente creado.

En primer lugar, para la función register se deberá verificar que el usuario no se encuentre registrado y además deberá ingresar un usuario y contraseña correctos, en este caso ambos serán un string, luego de verificar que esto se cumpla se creara el usuario como un objeto y se agregara a la lista de usuarios de la red social.

En segundo lugar, se tiene la función login, esta función necesita de un usuario y una contraseña, para desarrollar el login lo primero es recorrer la lista de usuarios registrados y ver si se encuentra en estos, de ser así se debe verificar que la contraseña dada sea la correcta, finalmente para completar la función se debe entregar el editor modificado con el usuario dentro de los usuarios activos.

Las siguientes funciones (create, share, add, rollback, revoke Access, search) son modificaciones de los documentos dentro del editor, por lo tanto, se deberá ingresar mediante los selectores y modificar distintos parámetros dependiendo de la función utilizada.

En el caso de la función visualize es para poder ver los cambios realizados dentro del editor, esta funciona con override.



Aspectos de Implementación

Para desarrollar el laboratorio utilicé el IDE “NetBeans 12.0” esto debido a que al investigar sobre el lenguaje encontré muchas fuentes y material implementados en este IDE, el sistema operativo fue Windows 10, y la versión de JKD es 11.0.12.

Los aspectos de mi implementación son los siguientes:

public class Usuario {

//Atributos

private static final AtomicInteger counter = new AtomicInteger();

private final int id;

private final Date fecha;

private final String Nombre;

private final String Contrasenia;

Como se puede observar lo primero fue definir la clase y luego implementar un id autoincremental, de esta misma manera se crea la clase documento.

Luego se creó la clase editor:

public class Editor{

private final String nombreEditor;

private final Date fecha;

List<Usuario> userlist;

List<Documento> doclist;

String activeUser;

Instrucciones de uso

El programa al ser un proyecto inicializado con gradle para ejecutarlo solo se quiere dentro del powershell de la carpeta escribir: .\gradlew.bat run

Luego de ejecutar el programa lo primero que nos pide es elegir si cargar datos dentro del editor colaborativo, o iniciar sin datos, luego nos muestra un panel en el cual nos podemos registrar, iniciar sesión o cerrar el programa, luego de iniciar sesión nos mostrara otro panel en el cual nos mostrara que queremos hacer, para elegir esta y las anteriores opciones simplemente tenemos que escribir el numero de la opción que deseamos, si elegimos una opción que no está nos volverá a mostrar el mismo panel nuevamente.

A continuación, se mostrarán imágenes a modo de ejemplo de lo anteriormente explicado:

Texto

Descripción generada automáticamente

Si elige la opción de iniciar con datos se recomienda utilizar la función visualize que esta en el siguiente panel de opciones, esto para conocer los usuarios y documentos ya creados.

Texto

Descripción generada automáticamente

Luego de elegir la opción tendremos que elegir si deseamos registrarnos o iniciar sesión, a modo de ejemplo se utilizara uno de los usuarios registrados previamente en el editor colaborativo.

Texto, Carta

Descripción generada automáticamente

Aquí finalmente tendremos todas las opciones que puede escoger un usuario una vez haya iniciado sesión.

Resultados y autoevaluación

El resultado de mi trabajo no es completo debido que no pude crear la totalidad de sus funciones, lo completado fue:

* Clases e interfaces100%
* Menú interactivo por terminal 100%
* Authetication 100%
* Create 100%
* Share 100%
* Add 100%
* Rollback 100%
* revokeAccess 100%
* Search 0%
* Visualize 100%

Dentro de los requerimientos no funcionales todos fueron completados en un 100%.

Conclusión

Para finalizar podemos decir que, gracias al conocimiento adquirido en cursos anteriores de C, se me hizo fácil el aprender cómo manejar las clases y sus métodos, esto debido al parecido de las clases con los struct de C, por otro lado, la forma en que opera este lenguaje es bastante intuitiva y fácil de aprender.

A modo de comparación con lenguajes anteriormente utilizados dentro de la asignatura, java es mucho más fácil de manejar, además la información dentro de internet y en videos tutoriales es mucho más fácil de encontrar.