Universidad de Santiago de Chile
Facultad de Ingeniería
Departamento de Ingeniería Informática

Paradigmas de programación

Laboratorio 3

Paradigma orientado a objetos

Christofer Rodríguez

Profesor: Víctor Flores Sánchez

Santiago – Chile
2-2020

Tabla de contenidos

Introducción	3
Descripción del problema	4
Descripción del paradigma	5
Análisis del problema	6
Diagrama de análisis	7
Diseño de la solución	8
Diagrama de diseño	10
Aspectos de la implementación	11
Instrucciones de uso	12
Resultados y autoevaluación	13
Conclusión	14
Referencias	15

Introducción

Stack Overflow es un famoso foro de preguntas y respuestas relacionadas, principalmente con la programación e informática; este foro cuenta con diferentes funcionalidades, tales como: preguntas, responder preguntas, votar, entre otras. En base a esto, es necesario hacer una simulación de un sistema de fotos tomando como referencia la funcionalidad de Stack Overflow en el lenguaje de programación Java, esto bajo el paradigma de programación orientado a objetos. En este documento primero se contextualizará con respecto al paradigma utilizado, se analizará el problema en cuestión, luego se explicará la solución creada para resolver este problema, para finalmente con los resultados obtenidos evaluar el grado de alcance de cada requerimiento.

Descripción del problema

Se necesita crea una representación de un foro de preguntas y respuestas basado en Stack Overflow, esto debe estar diseñado en el lenguaje de programación Java, utilizando el paradigma orientado a objetos. Esta representación debe contener características esenciales del foro como usuarios registrados, preguntas realizadas y respuestas a estas preguntas, por esto se necesita realizar una representación abstracta de este foro y confeccionar clases y métodos que simulan la interacción entre cada parte de esta abstracción, de esta manera se espera replicar el funcionamiento del foro original, en específico es necesario implementar el registro de un usuario en el foro, iniciar sesión, realizar preguntas, responder preguntas, ofrecer recompensas a quien responda mejor una pregunta, aceptar una respuesta como la mejor para una pregunta, votar a favor o en contra de una pregunta o respuesta y una manera de visualizar de una manera más grafica ciertas partes de esta simulación de foro cuando se requiera.

Descripción del paradigma

El paradigma de programación orientado a objetos o más conocido como POO, es una "forma" de programar en la cual se utiliza el concepto de objeto para simular o representar las entidades que se necesita emular, estos pueden poseer características que conoceremos como atributos, además de los métodos que corresponden al comportamiento de estos objetos.

Para crear estos objetos, se utiliza el concepto de clases, estas son la definición tanto de las características como los comportamientos que tendrá cierto tipo de objetos, mientras que los objetos serán una "instancia" de una clase, por esto, de una clase podremos crear múltiples objetos que pueden interactuar entre si u objetos de otras clases mediante diferentes métodos o relaciones entre ellos. Estos objetos se pueden tener diferentes tipos de relaciones, las cuales nos indican la manera en la que objetos de distintas clases se comunican entre sí, estas pueden ser de asociación, dependencia o herencia.

Como se nombró anteriormente la herencia es un tipo de relación muy importante en este paradigma, en ocasiones es necesario representar un grupo de entidades que poseen características en común o que derivan de una entidad más grande, la cual puede ser especifica o abstracta, para estos casos se utiliza la herencia, ya que esta nos permite "traspasar" características o comportamientos desde la entidad original o "más grande" a aquellas que deriven de ella, de esta manera se logra reutilizar o reducir código.

Ya que bajo este paradigma se busca la comunicación entre distintas entidades independientes, es esencial que bajo este paradigma se busque lograr una alta cohesión y bajo acoplamiento. Con esto se hace referencia a que cada representación de nuestro código debe representar una entidad en específico y enfocada solo en ella, además de lograr que cada una de estas entidades "conozca" solo lo necesario sobre el resto de las clases, de esta manera, un cambio en un lugar de nuestro código no implicará que distintas partes de él dejen de funcionar o deban ser modificadas.

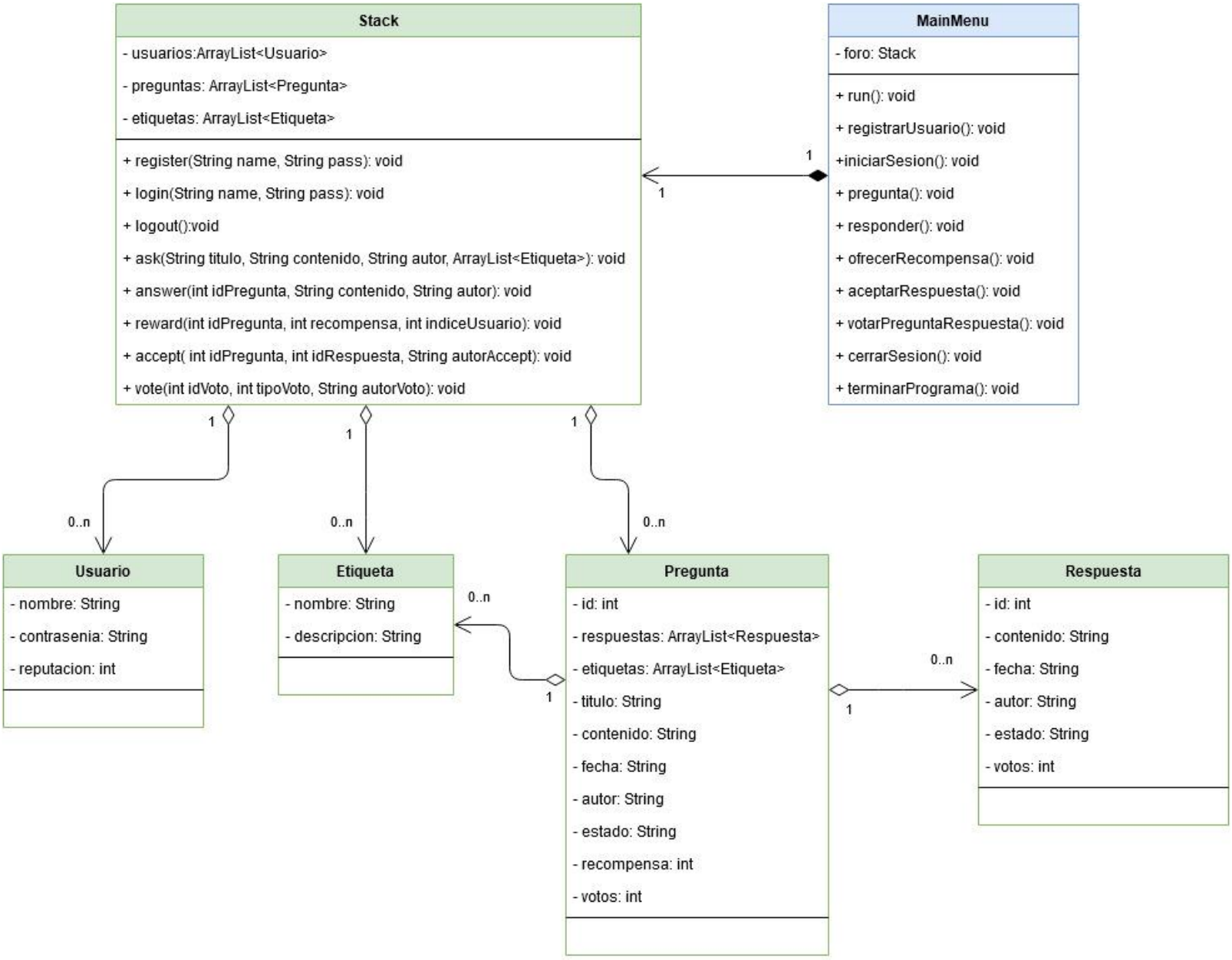
Análisis del problema

A diferencia del paradigma funcional y lógico estudiados anteriormente, los datos que creamos si pueden será almacenados y solamente actualizar su información cuando esto sea requerido.

Si analizamos los requerimientos para este problema, podemos notar que será necesario implementar una clase para representar el foro que almacenará los componentes de este, así mismo será necesario crea una clase para los principales componentes del foro, estos siendo preguntas, respuestas usuarios y etiquetas. Para la interacción con cada uno de estos componentes es necesario métodos para acceder, modificar o utilizar cada una de estas clases. De manera general será necesario crear métodos capaces de crear objetos, añadir nuevos elementos al registro, modificar valores de un elemento específico dentro de una lista, filtrar listas, verificar si es que podemos realizar ciertas funcionalidades, mostrar de manera visual al usuario componentes relevantes del foro según la acción que esté realizando, entre otros.

Además, será necesario que previamente se realice un diagrama de análisis para esta solución y una vez terminado el proceso de construir esta solución, se debe crear un diagrama de diseño que ilustre de manera correcta la implementación realizada.

Diagrama de análisis:



Diseño de la solución

Para llevar a cabo la simulación del foro, se utilizaron 10 clases diferentes los cuales serán detallados a continuación:

1. Usuario: Esta clase será la encargada de representar un usuario registrado dentro del foro, este contendrá un nombre, contraseña y reputación.

2. ListaUsuarios: Esta clase contendrá una lista de usuarios registrados dentro del foro.

3. Pregunta: Representación de una pregunta dentro del foro, almacenará la fecha de creación, su autor, titulo, contenido, etiquetas, identificador, estado, recompensas ofrecidas, votos y una lista de respuestas a esta pregunta.

4. ListaPreguntas: Esta clase contendrá una lista de preguntas creadas dentro del foro.

5. Respuesta: Representación de una respuesta registrada a una pregunta, contendrá el nombre de su autor, fecha de creación, contenido, votos, estado y su respectivo identificador.

6. ListaRespuestas: Esta clase contendrá las respuestas pertenecientes a una pregunta.

7. Etiqueta: Representará una etiqueta de una pregunta, esta solamente contendrá un identificador, su nombre y una breve descripción de ella.

8. ListaEtiquetas: Esta clase contendrá las etiquetas registradas dentro del foro o las etiquetas que representan una pregunta creada.

9. Stack: Esta clase será la representación del foro, “contendrá” a las clases anteriormente mencionadas con una lista de preguntas, una lista de usuarios registrados, una lista de etiquetas registradas en el foro y el nombre del usuario que inicio sesión, si es que se hizo.

10. MainMenu: Esta clase será la encargada de mostrar mensajes por consola al usuario, pedir que ingrese datos y entregar los datos ingresados por el usuario a los respectivos métodos que simularán el funcionamiento del foro.

Para respetar una alta cohesión y una mayor organización dentro del código se decidió representar cada entidad en las clases anteriormente mencionadas, además se decidió implementar una clase separada para una lista de preguntas, usuarios y respuestas, de esta manera se pueden separar los comportamientos que operan sobre una lista de estos objetos en específico de los métodos de la entidad que los contiene.

De manera general, para abordar este proyecto se decidió implementar una solución en la que de manera inicial se crea un Stack que contiene información pre registrada del foro y ya que bajo este paradigma se puede "guardar" la información y luego modificarla, se decidió realizar eso, cada vez que se necesitara modificar un aspecto del foro, se accedía a este y mediante su correspondiente método de modificaba esta información dentro del foro.

Los principales subproblemas presentes en la implementación de este proyecto se implementaron de la siguiente manera:

Interactuar con el usuario: Se implementó un menú principal que mostrará todas las opciones disponibles a realizar al usuario, este se encarga de mostrar todos los mensajes al usuario tales como: información relacionada a la acción a realizar, retroalimentación sobre la acción realizada y pedir que ingrese datos, finalmente este entregará la información ingresada a los correspondientes métodos.

Filtrar: El filtrar se utilizó específicamente para mostrar preguntas y/o respuestas que pertenezcan al usuario con sesión iniciada o que no pertenezcan a él, por esto se recorrieron las listas buscando que el nombre registrado en estas entradas fuera igual al nombre del usuario con sesión iniciada.

Verificar: Dentro de cada método, se comprueba que los datos ingresados sean correctos ya que en casos como en los que se filtran preguntas y/o respuestas, el usuario puede elegir algo que no se le haya mostrado, por esto se debe volver a verificar que pertenezca o no a él, además se verificó las características de los datos entregados para decidir qué acciones se realizarán.

Buscar: La búsqueda se utilizó de manera general en dos situaciones, la primera es comprobar la existencia de algo, en este caso se recorrió la lista de objetos a la vez que se comprobaba si dicho objeto contaba con las características buscadas. La segunda situación es cuando se necesitaba conocer el índice de un objeto dentro de una lista, al igual que la situación anterior se recorría la lista mediante iteración y cuando este objeto era encontrado se entregaba el numero de la iteración que representaba el índice dentro del objeto dentro de la lista que lo contiene, de esta manera se puede acceder directamente al objeto por su índice.

Modificar: Puesto de que casi la totalidad de la información modificable se encontraba contenida en una lista de objetos, cuando se necesitaba crear un objeto, este era creado y luego añadido al final de su correspondiente lista; en cambio cunado se necesitaba modificar la información de un elemento previamente existente, se procede a modificar mediante su identificador, ya que este difería en una unidad con su índice dentro de la lista para el caso de las preguntas y etiquetas, para el caso donde no se conocía el índice del elemento (usuarios) o donde su identificador no tenía relación con su posición dentro de la lista, mediante los métodos de búsqueda se obtenía su índice y se modificaba con su respectivo método.

Diagrama de diseño:

Aspectos de implementación

Para la implementación de este proyecto se utilizó IntelliJ como intérprete del lenguaje de programación Java utilizando la versión 11 de OpenJDK, solamente fueron utilizadas funcionalidades pertenecientes a la librería estándar de Java. Las clases utilizadas para la implementación de este programa fueron divididas en sus correspondientes package, si bien no se siguió de manera total el modelo MVC (modelo-vista-controlador), se creó el package para los modelos y el package para la vista del programa. Además, para la compilación del programa se creó un script de compilación ".bat", que realiza la compilación automática y posterior ejecución del programa.

Instrucciones de uso

Cabe mencionar que el siguiente proceso para compilar y ejecutar el programa fue probado en un computador con sistema operativo Windows, por lo que funciona en este ambiente, pero no se asegura que funciones en otros sistemas operativos.

Para poder utilizar el programa por primera vez es necesario compilar el código, esto se realiza ejecutando el archivo llamado “scriptCompilar.bat” que se encuentra en la carpeta del programa llamada “código fuente”, al ejecutarlo aparecerá una ventana emergente en la que luego de compilar los archivos necesarios, comenzará a ejecutarse el programa, mostrándole el menú principal.

En caso de que se quiera volver a utilizar el programa una vez cerrado la primera vez, se debe abrir el programa PowerShell dentro de la carpeta en la que se encuentran los archivos, luego dentro de la ventana emergente que se abrirá, se debe ingresar la siguiente línea de comando “java Main” y presionar la tecla “enter”. Si es que se quiere evitar el procedimiento anterior, igualmente puede volver a ejecutar el archivo “scriptCaompilar.bat”, pero esto volverá a compilar innecesariamente los archivos del programa, aunque funcionando igualmente.

Cabe señalar que cada vez que se ingresen datos al programa mediante la consola, es necesario pulsar la tecla “enter” para que se ingrese correctamente, además se aconseja ingresar solamente lo que señala el programa, ya que de no respetar estas instrucciones puede provocar que el programa deje de funcionar inesperadamente. Junto con lo anterior es necesario agregar que el programa entregará al usuario una retroalimentación sobre la acción realizada, esto quiere decir que se indicará cuando una acción de ejecutó correctamente o cuando no se pudo ejecutar correctamente, agregando también la razón de esto.

Resultados y autoevaluación

Se obtuvieron resultados satisfactorios al poner a prueba cada funcionalidad, para comprobar su correcto funcionamiento se realizaron consultas con distintos valores, en los casos de ingresar valores validos estas consultas funcionaron correctamente y los valores inválidos fueron detectados correctamente.

Clases y estructuras	Se logró implementar cada clase o entidad requerida para la solución del problema de manera satisfactoria y que permitiera una como trabajo.
Menú interactivo	Se logró implementar un menú interactivo que permitiera un fácil uso para el usuario y se implementó correctamente la información inicial del foro.
Register/Login/Logout	Se logro implementar cada una de estas funcionalidades, se decidió implementar cada una por separado para un más cómodo uso por parte del usuario.
Ask	Se creo de manera satisfactoria la funcionalidad para crear una nueva pregunta y se agregó la opción de crear nuevas etiquetas para las preguntas dentro de esta funcionalidad.
Answer	Se crea correctamente la respuesta del usuario y se muestran las preguntas con sus respectivas respuestas al usuario para que pueda elegir mejor que pregunta responder.
Reward	Se ofrece correctamente la recompensa a la pregunta, además se le muestra al usuario su reputación actual para que ofrezca. Debido a que en la implementación de esta solución no hay forma de retirar una recompensa ofrecida o que una pregunta sea eliminada, la recompensa ofrecida será inmediatamente descontada de la reputación y sumada a la recompensa de la pregunta.
Accept	Se implementó de manera satisfactoria, solo se muestran preguntas del usuario y solo se pueden aceptar preguntas creadas por este, además se actualiza el estado de la respectiva pregunta y respuesta, junto con la reputación de los usuarios involucrados.
Vote	Se modifica correctamente tanto los votos de la pregunta o respuesta y la reputación de los usuarios involucrados dependiendo del tipo de voto realizado.

Conclusión

Al igual que en el proyecto anterior, es difícil adaptarse a un nuevo paradigma y lenguaje, aunque ya que este paradigma es el más parecido al imperativo de entre los anteriormente estudiados, en ciertos momentos fue necesario volver a darle una vuelta a un subproblema para saber si era una correcta implementación para el paradigma orientado a objetos, pero el realizar previamente el diagrama de análisis permitió tener una vista más clara, tanto del problema como de la solución a desarrollar. Si bien este diseño de solución sirvió como lineamiento principal para el desarrollo de esta, fue necesario realizar cambios a este diseño, ya que al momento de programar la solución se pueden notar características que previamente no se consideraron o mejores maneras de implementar uno de los componentes. Debido a la poca costumbre con el lenguaje de programación Java, existieron ciertas limitaciones o una ineficiente utilización de la basta cantidad de funcionalidades de este lenguaje, aunque también sirvió para ver distintas soluciones a un problema. Por lo anteriormente descrito se considera que el proyecto fue realizado de una manera correcta y su desarrollo será de vital importancia para la formación como programador.

Referencias

Briceño, G. (2018). 10 conceptos de POO en Java. (Recuperado el 10/01/2021) https://www.clubdetecnologia.net/blog/2018/10-conceptos-de-poo-en-java/

Cevallos, K. (2015). UML: Relaciones entre clases. (Recuperado el 10/01/2021) https://ingsotfwarekarlacevallos.wordpress.com/2015/07/02/uml-relaciones-entre-clases/

Covantec R.L. (2020). 9.3. Programación orientada a objetos. (Recuperado el 10/01/2021) https://entrenamiento-python-basico.readthedocs.io/es/latest/leccion9/poo.html