PROGETTO DI PROGRAMMAZIONE 2

UNIVERSITÁ DI PISA
DIPARTIMENTO IN INFORMATICA
LAUREA IN INFORMATICA

ADRIEN KOUMGANG TEGANTCHOUANG
MATRICOLA

30 novembre 2020

Indice

1 Guida **2**

2 Class Post **2**
2.1 Ovveride 2
2.2 metodi 3

3 Interface SocialNetworkInterface **3**
3.1 Ovveride 3
3.2 metodi non statici 4
3.3 metodi statici 4

4 Class SocialNetwork **4**
4.1 Ovveride 4
4.2 metodi statici 5

5 Class SocialNetworkExtend **6**
5.1 Override 6
5.2 metodi non statici 6
5.3 metodi statici 6

6 Batterie di test **7**

1 Guida

Il progetto è diviso in vari package, di cui :

- **pubblication** : contenente la classe Post
- **my.app** : contenente l'interfaccia SocialNetworkInterface e le classe SocialNetwork e SocialNetworkExtend
- **my.all.exceptions** : contenente tutte le eccezioni non definite a proposito per ogni classe creata e non predefinita nel linguaggio
- **test** : contente le classe fatte per testare ogni metodo delle classe e interfacce definite

Le sorgente sono presente nella cartella **src**.

Nella cartella, è anche presente dei file html (formato java oracle) che presenta in modo semplice tutto il progetto intero. sono dentro la cartella **doc**. Ho anche allegato per ogni quel file, il formato pdf per poter accedere direttamente in formato pdf.

É stato allegato anche un file jpeg del diagramma UML delle classe, che modella la relazione fra le classi dichiarate.

2 Class Post

2.1 Ovveride

La classe Post è la classe che permette di modellare una pubblicazione fatta sulla rete sociale da un utente. Per questo, è caratterizzata da un identificatore univoco, dal nome dell'utente che l'ha pubblicato, dal contenuto della pubblicazione (un testo) e dalla data e ora d pubblicazione. Un oggetto di tipo Post è immutabile.

Typical element : un elemento tipico della classe Post è caratterizzato da

- il suo identificatore : id
- l'autore del post cioè l'utente che l'ha scritto e pubblicato : author
- il suo contenuto chi è un testo : text
- la data e giorno di pubblicazione (creazione) del post : timestamp
- l'insieme delle persone menzionate nel testo del post : mentioned

l'identificatore è formato dal nome dell'autore e dalla data di pubblicazione, così sono sicuro che sarà univoco.

Rep Invariant : per ogni istante $t > 0$

- this.id(t) = this.id(t+1) &&
- this.author(t) = this.author(t+1) &&
- this.text(t) = this.text(t+1) &&
- this.timestamp(t) = this.timestamp(t+1) &&
- this.mentioned(t) = this.mentioned(t+1) &&
- RI : id != null && author != null, "", " " && E != null && timestamp != null

2.2 metodi

- **equals(Post p)** : Metodo che ci permette di determinare se 2 istanze di Post sono uguali
- **getAuthor()** : Metodo che ci permette di avere il nome dell'utente chi ha fatto questa pubblicazione
- **getContent()** : Metodo che ci permette di avere il contenuto della pubblicazione (testo)
- **getId()** : Metodo che ci permette di avere l'identificatore univoco del post
- **getMentioned()** : Metodo che ci ritorna un iteratore sulla lista delle persone menzionate nel post
- **getPost()** : Metodo che mi permette di ritornare una forma direttamente stampabile del post
- **getTimeStamp** : Metodo che ci permette di avere la data e l'ora di pubblicazione de post sotto forma di stringa

3 Interface SocialNetworkInterface

3.1 Ovveride

L'interfaccia SocialNetworkInterface dà la definizione minimale di come deve essere una classe che lavora su una rete sociale definendo i metodi che devono avere tutti.

3.2 metodi non statici

- **containing(String words)** : Restituisce la lista dei post presenti nella rete che includono almeno una parola presenti nella lista delle parole argomento del metodo.
- **getMentionedUsers()** : Metodo che restituisce l'insieme degli utenti menzionati (inclusi) nei post presenti nella rete sociale.
- **writtenBy(String username)** : Restituisce la lista dei post effettuati dall'utente nella rete sociale il cui nome è stato dal parametro username

3.3 metodi statici

- **static getMentionedUser(List<Post> ps)** : Restituisce l'insieme degli utenti menzionati (inclusi) nella lista dei post. Questo metodo è statico perché non lavora su una istanza precisa di SocialNetwork.
- **static guessFollowers(List<Post>)** : Restituisce la rete sociale derivata dalla lista dei post (parametro del metodo). Questo metodo è statico perché non lavora su una istanza precisa di SocialNetwork.
- **static influences(Map<String, Set<String>> followers)** : Restituisce gli utenti più influenti della rete sociale (parametro del metodo), ovvero quelli che hanno un numero maggiore di "follower"
- **static writtenBy(List<Post> ps, String username)** : Restituisce la lista dei post effettuati dall'utente il cui nome è dato dal parametro username presenti nella lista ps. Questo metodo è statico perché non lavora su una istanza precisa di SocialNetwork

4 Class SocialNetwork

4.1 Ovveride

La classe SocialNetwork modella una rete sociale dove delle persone possono pubblicare dei post e gli altri possono interagire soppra e possono essere anche identificate dentro quelli post.

Typical element : un elemento tipico della classe SocialNetwork è caratterizzato da :

- **users** : gli utenti registrati nella rete sociale
- **reteSociale** : mappa descrivendo collegamenti fra persone

- **posts** : insieme dei post presenti nella rete sociale e fatta da utenti registrati
- **likes** : mappa che associa ad ogni post, l'insieme degli utenti che hanno messo mi piace al post

Rep Invariant :

- post $\in$ this.posts $\rightarrow$ post.author $\in$ users &&
- user $\in$ this.likes $\rightarrow$ user $\in$ this.users &&
- this.reteSociale[user] = {u | u $\in$ this.likes[post.user] or $\exists$p | u mentioned in p with p.Auhtor = user}

4.2 metodi statici

Questi metodi sono statici perché non hanno effetto diretto sullo stato di un tipico elemento di tipo SocialNetwork e quindi sono indipendenti di esso.

- **addLike(Post p, String name)** : Metodo che permette di aggiungere dei like ad un post.
- **addPost(Post p)** : Metodo che permette di aggiungere un post alla rete sociale e aggiorna la rete sociale aggiungendo, se non ci sono già, le persone menzionate nel post come i followers dell'autore del post.
- **addUsers(string user)** : Metodo che permette di aggiungere un utente alla rete sociale
- **containing(List<String> words)** : Restituisce la lista dei post presenti nella rete sociale che includono almeno una delle parole presenti nella lista delle parole presente nell'argomento del metodo.
- **getAllPost()** : Ritorna una lista contenente tutti i post presente nella rete sociale.
- **getMentionedUsers()** : Restituisce l'insieme degli utenti menzionati (inclusi) nei post presenti nella rete sociale.
- **getUsers()** : Metodo che ci permette di ritornare la lista degli utenti presenti nella rete sociale sotto forma di iteratore
- **removeAllPost(Collection¡Post¿ cp)** : Rimuove un insieme di post della rete sociale
- **removePost(Post p)** : Rimuove un post della rete sociale
- **String (username)** : Restituisce la lista dei post effettuati dall'utente nella rete sociale il cui nome è dato dal parametro username

5 Class SocialNetworkExtend

5.1 Override

La classe SocialNetworkExtend è un estensione della classe SocialNetwork in cui si gestisce la segnalazione dei post possibilmente contenendo delle parole offensive.

L'idea è quella che, c'è un'insieme di parole che si può aggiornare ad ogni momento, contenente delle parole dette offensive di cui, ogni volta che va fatto una pubblicazione, si controlla che può avere o meno delle parole offensive. Ed è possibile segnalare un post come potenzialmente offensive.

Typical Element : un elemento tipico della classe SocialNetworkExtend è caratterizzato :

- **super.Typical Element** : elemento tipico del suo padre SocialNetwork
- **reportedPost** : insieme dei post contenendo del contenuto abusivo

Rep Invariant :

- RI del padre &&
- p in this.reportedPost $\rightarrow \exists w \in p$ tale che $w \in this.badWords$

5.2 metodi non statici

- **addPost(Post p)** : Metodo che mi permette di aggiungere un post alla rete sociale
- **reportedPost(Post p)** : Metodo che mi permette di segnalare un post come contenente del contenuto offensivo o abusivo

5.3 metodi statici

- **addWord(string)** : Metodo che mi permette di aggiungere una parola all'insieme delle parole dette offensive
- **addWords(Collection<String> words)** : Metodo che mi permette di aggiungere un insieme di parole all'insieme delle parole offensive
- **containtBadWords(Post p)** : Metodo privato che mi permette di determinare se il post contiene delle parole offensive
- **getBadWords()** Metodo che permette di avere un iteratore sull'insieme delle parole offensive

- **removeWord(String word)** : Metodo che mi permette di togliere una parola nell'insieme delle parole dette offensive
- **removeWords(Collection<String> words)** : Metodo che mi permette di togliere un insieme di parole all'insieme delle parole offensive

6 Batterie di test

- **TestPost** : per testare la classe Post
- **TestSocialNetwork** : per testare la classe SocialNetwork
- **TestSocialNetworkExtend** : per testare la classe SocialNetworkExtend
- **TestSocialNetworkInterface** : per testare i metodi statici dell'interfaccia SocialNetworkInterface