Code for Facebook

Write object-oriented code to implement the design of the Facebook problem.

We'll cover the following

We've reviewed different aspects of Facebook and observed the attributes attached to the problem using various UML diagrams. Let's now explore the more practical side of things, where we will work on implementing the Facebook network using multiple languages. This is usually the last step in an object-oriented design interview process.

We have chosen the following languages to write the skeleton code of the different classes present in Facebook:

- Java
- C#
- Python
- C++
- JavaScript

Facebook classes

In this section, we will provide the skeleton code of the classes designed in the class diagram lesson.

> **Note:** For simplicity, we are not defining getter and setter functions. The reader can assume that all class attributes are private and accessed through their respective public getter methods and modified only through their public method functions.

Constants

The following code provides the definition of the various enums and custom data types being used in the Facebook design:

> **Note:** JavaScript does not support enumerations so we will be using the `Object.freeze()` method as an alternative that freezes an object and prevents further modifications.

```java
public class Address {
  private int zipCode;
  private String houseNo;
  private String city;
  private String state;
  private String country;
}

enum AccountStatus {
  ACTIVE,
  BLOCKED,
  DISABLED,
  DELETED
}

enum FriendInviteStatus {
  PENDING,
  ACCEPTED,
  REJECTED,
  CANCELED
}

enum PostPrivacySettings {
  PUBLIC,
  FRIENDS_OF_FRIENDS,
  ONLY_FRIENDS,
  CUSTOM
}
```

Constant definitions

Interfaces implemented by user

Facebook will have several interfaces that will be implemented by users and are described below:

- `PageFunctionsByUser`: This will define the functions that a user will perform while interacting with pages.
- `GroupFunctionsByUser`: This will define the functions that a user will perform while interacting with groups.
- `PostFunctionsByUser`: This will define the functions that a user will perform while interacting with posts.
- `CommentFunctionsByUser`: This will define the functions that a user will perform while interacting with comments.

The definition of all these interfaces is given below:

```java
public interface PageFunctionsByUser {
  public Page createPage(String name);
  public Page sharePage(Page page);
  public void likePage(Page page);
  public void followPage(Page page);
  public void unLikePage(Page page);
  public void unFollowPage(Page page);
}

public interface GroupFunctionsByUser {
  public Group createGroup(String name);
  public void joinGroup(Group group);
  public void leaveGroup(Group group);
  public void sendGroupInvite(Group group);
}

public interface PostFunctionsByUser {
  public Post createPost(String content);
  public Post sharePost(Post post);
  public void commentOnPost(Post post);
  public void likePost(Post post);
}

public interface CommentFunctionsByUser {
  public Comment createComment(Post post, String content);
  public void likeComment(Comment comment);
}
```

The interfaces present in Facebook

Account

The `Account` class refers to an account of a user on Facebook and is responsible for containing their personal details, such as username, password, etc. It also allows users to reset their existing passwords. The definition of this class is given below:

```java
public class Account {
  private String accountId;
  private String password;
  private String username;
  private String email;
  private AccountStatus status;

  public boolean resetPassword();
}
```

The Account class

Person, user, and admin

The `Person` class will be an abstract class that represents a normal Facebook user. A `User` can also be an `Admin`. The definition of these two classes is provided below:

> **Note:** The `User` class will implement the interfaces mentioned above. A few of these have been mentioned in the code below.

```java
public class Admin {
  public void blockUser(User user);
  public void unblockUser(User user);
  public void enablePage(Page page);
  public void disablePage(Page page);
  public void deleteGroup(Group group);
  public void changeGroupPrivacy(Group group);
}

// Person will be an abstract class
public abstract class Person {
  private String name;
  private Address address;
  private String email;
  private String phone;
  private Account account;
}

// We will be using only one interface example
public class User extends Person implements PageFunctionsByUser{
  private int userId;
  private String name;
  private Date dateOfJoining;
  // The following lists contain the pages and groups that a user is admin of
  private List<Pages> pagesAdmin;
  private List<Groups> groupsAdmin;

  private Profile profile;
  public boolean sendMessage(Message message);
  public boolean sendRecommendation(Page page, Group group, User user);
```

The Person, User and Admin classes

Profile, education, places, and work

The `Work`, `Education`, and `Places` classes will provide a user's personal information and will make up the `Profile` class. The definition of these classes is given below:

```java
public class Profile {
  private String gender;
  private byte[] profilePicture;
  private byte[] coverPhoto;
  private List<User> friends;

  private List<int> usersFollowed;
  private List<int> pagesFollowed;
  private List<int> groupsJoined;

  private List<Work> workExperience;
  private List<Education> educationInfo;
  private List<Place> places;

  public boolean addWorkExperience(Work work);
  public boolean addEducation(Education education);
  public boolean addPlace(Place place);
  public boolean addProfilePicture(byte[] image);
  public boolean addCoverPhoto(byte[] image);
  public boolean addGender(String gender);
}

public class Work {
  private String title;
  private String company;
  private String location;
  private String description;
  private Date startDate;
  private Date endDate;
}
```

The Profile, Education, Places and Work classes

Page, post, and comment

Facebook users can create and like pages, posts, and comments. The definition of these classes is given below:

```java
public class Page {
  private int pageId;
  private String name;
  private String description;
  private int likeCount;
}

public class Post {
  private int postId;
  private String content;
  private byte[] image;
  private int likeCount;
  private int shareCount;
  private User postOwner;
  private PostPrivacySettings settings;
  public changePostVisibility(Post post);
}

public class Comment {
  private int commentId;
  private String content;
  private int likeCount;
  private User commentOwner;
}
```

The Page, Post, and Comment classes

Profile privacy

Each profile will have its own privacy settings where users can change the visibility settings of their friends' list, and lock their profile and profile picture.

```java
public class ProfilePrivacy {
    public void changeFriendsListVisibility(Profile profile) {}
    public void lockProfile(Profile profile) {}
    public void lockProfilePicture(Profile profile) {}
}
```

The ProfilePrivacy class

Group and the group functions interface

Facebook users can also create and join groups as well as add new and delete new users. In addition, users that initially join groups will be notified of all new activities in the groups. The definition of these classes is given below:

```java
public interface GroupFunctions {
  public boolean addUser(User user);
  public boolean deleteUser(User user);
  public boolean notifyUser(User user);
}

public class Group implements GroupFunctions {
  private int groupId;
  private String name;
  private String description;
  private byte[] coverPhoto;
  private int totalUsers;
  private boolean isPrivate;
  private List<User> users;

  public boolean addUser(User user) {
    // functionality
  }
  public boolean deleteUser(User user) {
    // functionality
  }
  public boolean notifyUser(User user) {
    // functionality
  }
  public void updateDescription(String description) {}
  public void addCoverPhoto(byte[] image) {}
}
```

The Group class and the GroupFunctions interface

Message and friend request

Each Facebook user can send messages and friend requests to other users. The definition of both of these classes is given below:

```java
public class Message {
  private int messageId;
  private User sender;
  private String content;
  private List<User> recipient;
  private List<byte[]> multimedia;

  public boolean addRecipent(List<User> users);
}

public class FriendRequest {
  private User recipent;
  private User sender;
  private FriendRequestStatus status;
  private Date requestSent;
  private Date requestStatusModified;

  public boolean acceptRequest(User user);
  public boolean rejectRequest(User user);
}
```

The Message and FriendRequest classes

Notification

The `Notification` class is responsible for sending notifications to users about any new messages, comments, posts, or friend requests via the built-in notification option. Its definition is given below:

```java
public class Notification {
  private int notificationId;
  private Date createdOn;
  private String content;

  public boolean sendNotification(Account account);
}
```

The Notification class

Search catalog and interface

The `SearchCatalog` class contains information on members, groups, pages, and various posts. It also implements the `Search` interface class to enable the search functionality based on the given criteria (member, group, page names, and post keywords). The definition of these two classes is given below:

```java
public interface Search {
  public List<User> searchUsers(String name);
  public List<Group> searchGroups(String name);
  public List<Page> searchPages(String name);
  public List<Post> searchPosts(String keywords);
}

public class SearchCatalog implements Search {
  HashMap<String, List<User>> userNames;
  HashMap<String, List<Group>> groupNames;
  HashMap<String, List<Page>> pageTitles;
  HashMap<String, List<Post>> posts;

  public boolean addNewUser(User user) {}
  public boolean addNewGroup(Group group) {}
  public boolean addNewPage(Page page) {}
  public boolean addNewPost(Post post) {}

  public List<User> searchUsers(String name) {
    // functionality
  }

  public List<Group> searchGroups(String name) {
    // functionality
  }

  public List<Page> searchPages(String name) {
    // functionality
  }

```

The Search interface and Search Catalog class

Wrapping up

We've explored the complete design of Facebook in this chapter. We've looked at how Facebook can be visualized using various UML diagrams and designed using object-oriented principles and design patterns.

← Back

Activity Diagram for Facebook

Complete

Next →

Getting Ready: An Online Stock Brokerage System