Email Application Project Documentation

Table of Contents

1. Introduction

The Email Application is a Java program designed to simulate basic email functionalities. It allows users to create an email account, manage mailbox capacity, set an alternate email, change passwords, and send emails. The program is implemented in Java and utilizes object-oriented programming principles.

2. Code Overview

The Java code consists of the following main components:

- **EmailApplication** class: This class encapsulates the main functionalities of the email application, including account creation, password generation, and email sending.
- **generateRandomPassword** method: This method generates a random password based on a given length.
- Various setter methods: These methods enable the user to set mailbox capacity, an alternate email, and change the password.
- **sendEmail** method: This method facilitates the sending of emails to recipients.
- User-friendly interface: The code includes a menu-driven interface to interact with the email application.

3. Class Structure

The ***EmailApplication*** class has the following structure:

- **Attributes:**
- **firstName**: Stores the first name of the user.
- **lastName**: Stores the last name of the user.
- **password**: Stores the password for the email account.
- **defaultMailboxCapacity**: Represents the default mailbox capacity, set to 500 by default.
- **alternateEmail**: Stores an alternate email address.
- **email**: Represents the generated email address.
- **companySuffix**: Represents the company suffix for the email, set to "gmail.com" by default.
- **Constructor:**
- **EmailApplication(String firstName, String lastName)**: Initializes the **EmailApplication** object with the provided first and last names.
- **Methods:**
- **generateRandomPassword(int length)**: Generates a random password of the specified length.
- **setMailboxCapacity(int capacity)**: Sets the mailbox capacity to the specified value.
- **setAlternateEmail(String altEmail)**: Sets the alternate email address.
- **changePassword(String newPassword)**: Allows changing the password after validation.
- **isPasswordStrong(String password)**: Checks whether the provided password is strong.
- **sendEmail(String recipient, String message)**: Sends an email to the specified recipient.
- **Main Method:**
- **main(String[] args)**: Acts as the entry point for the program and provides the user interface for interacting with the email application.

4. Functionality

- **Email Creation:**
- The code allows users to create an email account based on their first name and last name. It generates an email address by combining the lower-cased versions of the first and last names with a predefined company suffix.

- **Password Generation:**
- It automatically generates a random password for the newly created email account. The password is a combination of uppercase and lowercase letters, numbers, and special characters.
- **Mailbox Capacity Management:**
- Users can set the mailbox capacity as per their requirements. The default mailbox capacity is set to 500MB, but users can modify it according to their needs.
- **Alternate Email Configuration:**
- Users can provide an alternate email address for the account. The code sets the alternate email by combining the user-provided address with the company suffix.
- **Password Change:**
- Users can change their passwords for the email account. The code enforces password strength criteria, requiring the password to contain uppercase letters, lowercase letters, numbers, and special characters. If the new password meets the criteria, it updates the password; otherwise, it prompts the user to choose a stronger password.
- **Email Sending:**
- The code provides a method to send emails. It takes the recipient's email address and a message as input and simulates sending an email by printing the recipient's email address along with the message to the console.
- **User-Friendly Interface:**
- The code implements a simple menu-based interface for users to interact with the email application. Users can choose from options such as setting an alternate email, adjusting mailbox capacity, changing the password, sending emails, and exiting the application.
- **Error Handling:**
- The code includes exception handling to catch and display any errors that might occur during the program's execution. It ensures that the application doesn't terminate abruptly and provides a graceful error message to the user.

5. Usage

Upon running the application, users will be prompted to provide their first name and last name. The program will generate a unique email address and a secure password for the user. Users can then interact with the program through a simple menu-driven interface.

The menu options include:

- Set Alternate Email: Allows users to set an alternate email address.
- Set Mailbox Capacity: Enables users to specify the mailbox capacity.
- Change Password: Facilitates the modification of the account password.
- Send Email: Allows users to send emails to specified recipients.
- Exit: Terminates the program.

Users can select the desired option by entering the corresponding number. The program will execute the selected action and prompt the user for any required input.

6. Conclusion

The Email Application project provides a simple and functional email management system with password generation and validation. It demonstrates object-oriented programming principles and user interaction through the command line.

7. Appendix

Source code:

Author: *Shweta Gaikwad*

Date: *15 Oct 2023*