Java Software Development

Homework 5

Class Design

- Define a class named `Document` that contains an instance variable of type `String` named `text` that stores any textual content for the document.
- Create a method named `toString` that returns the `text` field.
- Create a mutator method for setting the value of the `text` field.

Document
text: String
+ toString(): String + setText(String): void

Class Design

- Next, define a class for `Email` that is derived from `Document` and includes instance variables for the `sender`, `recipient`, and `title` of an email message. The body of the email message should be stored in the inherited variable `text`.
- You should implement appropriate accessor and mutator methods for the class fields. Redefine the `toString` method to concatenate all text fields.

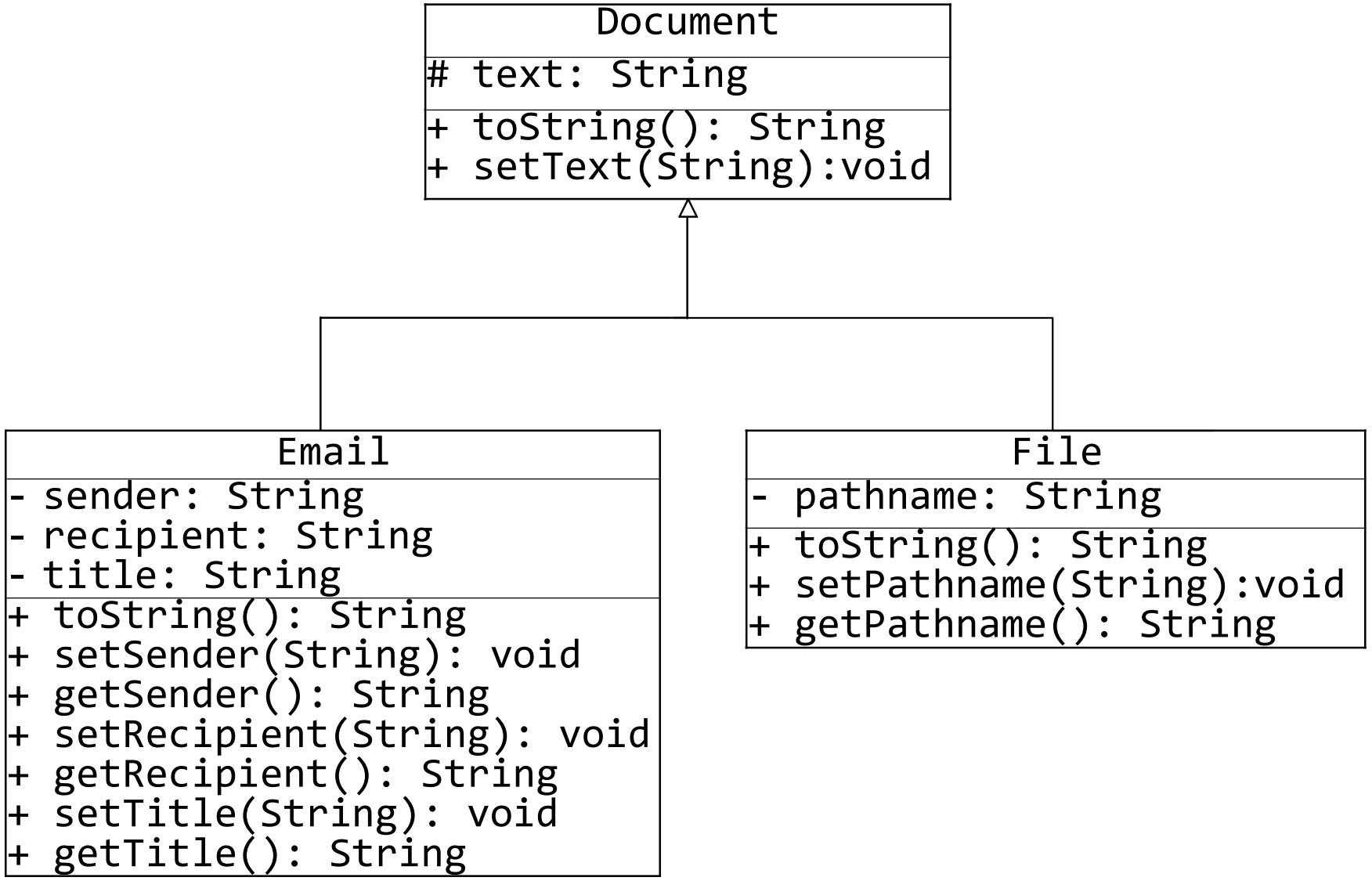
Email
- sender: String - recipient: String - title: String
+ toString(): String + setSender(String): void + getSender(): String ...

Class Design

- Similarly, define a class for `File` that is derived from `Document` and includes a instance variable for the `pathname`. The textual contents of the file should be stored in the inherited variable `text`.
- Implement appropriate accessor and mutator methods. Redefine the `toString` method to concatenate all text fields.

File
- pathname: String
+ toString(): String + setPathname(String): void + getPathname(): String

Class Design



Text Format

- `Document.toString()`

```
text
```

- `Email.toString()`

```
From: sender
To: recipient
Title: title
text
```

- `File.toString()`

```
Path: pathname
text
```

Problem Description

- Write a program to print some information about a document according to the execution mode.
- The program input is given from keyboard
  - Input 1: The execution mode (`A`, `B` or `C`)
  - Input 2: The document type (`Document`, `Email` or `File`)
- For type `Document`:
  - Input 3: document content
- For type `Email`:
  - Input 3: sender
  - Input 4: recipient
  - Input 5: title
  - Input 6: email body
- For type `File`:
  - Input 3: path to the file
  - Input 4: file content

Problem Description

- For execution mode `A`, you should use the `toString()` method to print the text content of the document.
- For execution mode `B`, enter a keyword that follows the last keyboard input. You should print whether or not the text content contains the keyword (print true or false).
- For execution mode `C`, enter one of the name of an instance variable and the new value of the variable. You should update the value of the variable and print all content.
  - For type `Document`, the variable name must be "text"
  - For type `Email`, the variable name must be "text", "sender", "recipient" or "title", one of above.
  - For type `File`, the variable name must be "text" or "pathname", one of above.

Sample Input and Output

Input 1	A
Input 2	Email
Input 3	Rose
Input 4	Jack
Input 5	Titanic
Input 6	I will always love you!
Output	From: Rose To: Jack Title: Titanic I will always love you!

Sample Input and Output

Input 1	`B`
Input 2	`Document`
Input 3	`Friday is a delightful day.`
Input 4	`Monday`
Output	`false`

Sample Input and Output

Input 1	C
Input 2	File
Input 3	D:\java\final_exam.docx
Input 4	YOU CANNOT PASS!!!
Input 5	text
Input 6	I will pass!!!
Output	Path: D:\java\final_exam.docx I will pass!!!

Submission

- Please archive your source code to `STUDENT_ID.zip` (download the example zip file from Moodle) and upload to Moodle before deadline.
- Your zip file should follow the following format.
  - `STUDENT_ID.zip`

    ```
    |- src
    |- META-INF
    |    |- MANIFEST.MF
    ```

  - All the source files (*.java) are put in the src directory.
  - The entry point (i.e. main class) of the program is specified in the MANIFEST.MF file.
- No late submission is accepted.