Validation of Email, Username and Password using JAVA

Group 5 | LI NANKAN, CHANG LIU

Java project

2023

Contents

[o Explication of the planning and development of the code 2](#_Toc129170311)

[o Basic descriptive process of each method 2](#_Toc129170312)

[isAlphaNum() 2](#_Toc129170313)

[isSpecialChar() 2](#_Toc129170314)

[isPrefixChar() 2](#_Toc129170315)

[isDomainChar() 2](#_Toc129170316)

[singleAtSign() method 3](#_Toc129170317)

[fetchBeforeAt() method 3](#_Toc129170318)

[fetchAfterAt() method 3](#_Toc129170319)

[isPrefix() method 3](#_Toc129170320)

[isDomain() method 3](#_Toc129170321)

[isEmail() method 3](#_Toc129170322)

[isUsername () method 3](#_Toc129170323)

[safePassword() method 3](#_Toc129170324)

[o Flow charts 4](#_Toc129170325)

[o Problems that arose during the development of the website 4](#_Toc129170326)

[regular expression 4](#_Toc129170327)

[o Individual team assignments 4](#_Toc129170328)

[o Overview of the project, workload, difficulty, comprehension 4](#_Toc129170329)

# o Explication of the planning and development of the code

We mainly have three methods to validate the Email, the Username and the Password respectively. We plan to break the three different methods to several small methods firstly, e.g., the Email validation method can be broken to singleAtSiign() method which checks if a String contain a single at sign(@). If true, we can then develop two methods named fetchBeforeAt() and fetchAfterAt() to get the string before @ and after @. Then we can develop two different methods isPrefix() and isDomain() to check if the string before @ is a valid Email prefix and string after @ is a valid Email domain. We can also break down the isPrefix() and isDomain() methods to isPrefixChar() and isDomainChar() which checks if the character is allowed in the prefix or in the domain or not. Since these two methods both need to check if a character is alphanumeric or special character. We can develop two different methods to judge: isAlphaNum() and isSpecialChar().

After the different steps of the programme is created, we can write the different methods from the basic to more complicated ones. The complicated methods will call the basic methods to implement its function, e.g., the isEmail() method will call singleAtSign() method if true, then call isPrefix() to check the prefix of the Email, then call isDomain() to check the domain of the Email.

# o Basic descriptive process of each method

## isAlphaNum()

This method will check if a character is alphanumeric. The method isAlphaNum() takes a char as

its only argument. Return true if the character is a letter of the English alphabet (case insensitive) or a number between 0 and 9, false otherwise.

## isSpecialChar()

This method will check if a character is an acceptable special character. The method

isSpecialChar() takes a char, and boolean as its argument. The char will represent the character to

validate and the boolean indicates if the underscore is considered a special character.

Return true if such character is a dash (-), period (.), if the boolean argument is true also allow for the

underscore (\_) to return true, false otherwise.

## isPrefixChar()

The is PrefixChar() checks if a character is a character allowed in the prefix. The method

isPrefixChar() takes a char as its only argument. Return true if the character can be used in the prefix, false otherwise. It will call isAlphaNum() and isSpecialChar() methods to check the character.

## isDomainChar()

The isDomainChar() checks if a character is a character allowed in the domain. The method

isDomainChar() takes a char as its only argument. Return true if the character can be used in the domain (second portion), false otherwise. It will call isAlphaNum() and isSpecialChar() methods to check the character.

## singleAtSign() method

The singleAtSign() method check if a String contain a single at sign (@). The method singleAtSign() takes a String representing a possible email address as its only argument. Return true if the string contains exactly one '@', false otherwise.

## fetchBeforeAt() method

The fetchBeforeAt() method is too get the beginning of an email address. It takes a String

as its only argument. Return a String containing the portion before the @ symbol.

## fetchAfterAt() method

Similarly, the fetchAfterAt() method gets the ending of an email address. It also takes a String as its

only argument. Return a String containing the portion after the @ symbol.

## isPrefix() method

The isPrefix() method is to check if the start of a String is a valid email prefix. It takes a

String as its only argument. Return true if some conditions are met, false otherwise. It firstly calls fetchBeforeAt() method to get the prefix of a email. Then it calls isPrefixChar() to check if every character in the prefix is valid, which is what isPrefixChar() method does.

## isDomain() method

The isDomain() method is to check if the end of a String is a valid email domain. It takes a

String as its only argument. Return true if some conditions are met, false otherwise. It firstly calls fetchAfterAt() method to get the domain of a email. Then it calls isDomainChar() to check if every character in the domain is valid, which is what isDomainChar() method does.

## isEmail() method

The isEmail method is used to check if a string is a valid email address. Since the most obvious and basic character of a email address is that it contains @ symbol. So we can create a submethod to do the job, e.g., singleAtSign(). If the basic condition is met, then we can judge if the prefix and domain of the email is valid. Before judging, we need to fetch the prefix and domain parts of the email using two methods: fetchBeforeAt() and fetchAfterAt(). After the two parts are fetched, we can create submethods to judge, e.g., isPrefix() to check the prefix part and isDomain() to check the domain part.

## isUsername () method

we used the isAlphaNum(), isSpecialChar() and isDomainChar() to decide whether the elements of the username are acceptable and in right formats.

## safePassword() method

We used isSpecialChar() to check the elements in the password for the acceptable special characters. And use the flags of capitalcase , lowercase and numbers to check the password . We use a splitting and counting method to make sure the password should reach the last requirement of assignment.

# o Flow charts

Diagram

Description automatically generated

Figure flow chart of programme

# o Problems that arose during the development of the website

## regular expression

For the method of safePassword(), we plan to use regular expression to test if the same character is repeated more than twice in a row. At first, we do not know how to express the same character more than twice in a row. By searching on Internet, we found that capturing groups are a way to treat multiple characters as a single unit. They are created by placing the characters to be grouped inside a set of parentheses. For example, the regular expression (dog) creates a single group containing the letters "d", "o", and "g".

# o Individual team assignments

Each team member is responsible for creating some parts of the whole programme and also the corresponding part of document.

# o Overview of the project, workload, difficulty, comprehension

It is a very interesting assignment for both of us. It does help us to know better of Email which we use everyday. The workload is ok as we can write some basic codes and reuse them multiple times. We met some difficulties when we did the last part of the assignment which is the codes of the safePassword part. Because tow of us has the different understanding of the last requirement. It cost us a much long time to finish it. Anyway, we did it and learn something about Email.