AshLCorse / Regex Email Pattern Tutorial

Created 3 days ago

Code Revisions 1 Embed <script src="https://g Download ZIP

Regex Email Pattern Tutorial for DU's edX bootcamp

Regex Email Pattern Tutorial

```
# Understanding the Email Regex Pattern

In this tutorial, lets explore a commonly used regular expression (regex) pattern designed to validate email addresses. By the end of t

## Regex Summary

The regular expression we are working with is:

/^([a-z0-9_\.-]+)@([\da-z\.-]+)\.([a-z\.]{2,6})$/

This pattern ensures that the input string matches the structure of a valid email address, which includes a username, followed by the '

## Table of Contents

- [Anchors](#anchors)
- [Character Sets](#character-sets)
- [Quantifiers](#quantifiers)
- [Grouping And Capturing](#grouping-and-capturing)
- [Escaping Special Characters](#escaping-special-characters)
- [The OR Operator](#the-or-operator)
- [Flags](#flags)
- [Character Escapes](#character-escapes)
- [Character Classes](#character-classes)

## Regex Components
```

```

### Anchors: ^ and $

^: The ^ symbol signifies the _start_ of the string. It ensures that the regex pattern matches from the very start of the string.

$: The $ symbol signifies the _end_ of the string. It ensures that the entire string must match the regex pattern, which prevents any 

For example:

For ash.123@gmail.com, this ensures that the entire email is validated without any extra characters.

### Character Sets: [a-z0-9_\.-]

This character set allows lowercase letters (a-z), digits (0-9), underscores (\_), dots (.), and hyphens (-) in the email’s username ar

\d: Within the domain part, \d allows digits.

For example:

In the email ash.123@gmail.com, the username and domain can include letters, digits, and special characters like - and ..

### Quantifiers: + and {2,6}

+: The + quantifier means "one or more" of the preceding character set, ensuring that the username and domain have at least one charact

{2,6}: This quantifier specifies a _range_. The {2,6} following [a-z\.] means the top-level domain (like .com or .org) must be between 

For Example:

In ash.123@gmail.com, the .com part is between 2 to 6 characters, which matches the {2,6} requirement.

### Grouping And Capturing: ([a-z0-9_\.-]+) and ([\da-z\.-]+) and ([a-z\.]{2,6})

The regex uses capturing groups to match different parts of the email (username, domain, and top-level domain).
Parentheses () around [a-z0-9_\.-]+ capture the _username_ part, while ([\da-z\.-]+) captures the _domain_, and ([a-z\.]{2,6}) captures

Example:
```

```

In the email ash.123@gmail.com, ash.123 is captured by the first group, gmail by the second, and com by the third.

### Escaping Special Characters: \. and @

\.: Since the dot . is a special character in regex (representing any character), we escape it with a backslash \. to literally match a

@: The @ symbol is matched directly in the regex, ensuring it is present between the username and domain.

Example:

In ash.123@gmail.com, the @ and . must appear exactly as written, with no substitutions.

### The OR Operator: [] [^...]

[] Matches a character that is contained within the brackets.

e.g. in our section of code:

[a-z] Matches the range of lower case letters from "a" to "z".

### Flags:

Regex flags modify the behavior of a pattern, such as making it case-insensitive (i), global for all matches (g), or treating . as matc

### Character Escapes:

Character escapes represent non-printable characters or special characters that don't have a specific printable form. These are predefi

For example:

\d matches any digit ([0-9]).
\w matches any word character (alphanumeric or underscore, equivalent to [a-zA-Z0-9_]).
\n matches a newline.

### Character Classes: \d \w

```

```
\d Matches a single character that is a digit.

\w Matches a word character.

## Author

[Ashleigh Corse - Junior Developer](https://github.com/AshLCorse)

I am a Software Development student at DU's edX Full Stack Flex Coding bootcamp.
```