Regular Expressions in Java Notes

Team Treehouse

May 28, 2020

- Resources
  - **Java Pattern Documentation:** link
  - **Online Java Regex tester:** link
- *STRING_VAR.matches(...)*
  - Tells if the string matches the given regular expression
  - Is equivalent to '*R̂EGEX_EXPRESSION$*'
  - Can be used to validate something

  Example:

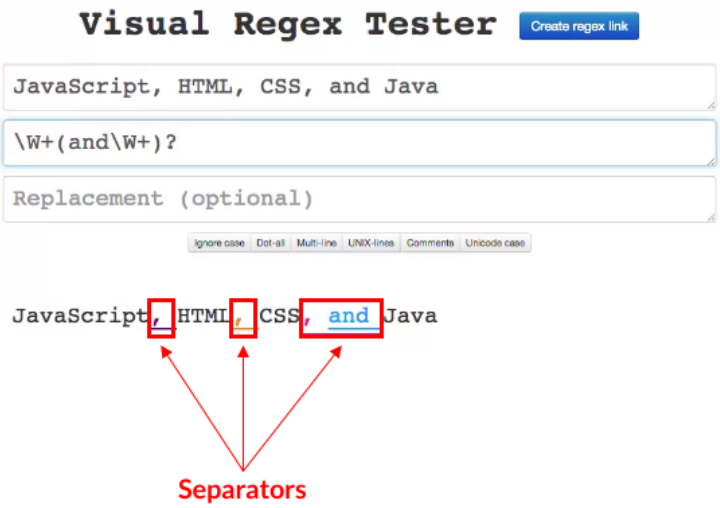
```
    Console console = System.console();
    String zipCode = "90210";
    if (zipCode.matches("^\\d{5}(-\\d{4})?$")) {
    System.out.printf("%s is a valid zip code%n", zipCode);
    } else {
    System.out.printf("%s is NOT a valid zip code%n", zipCode);
    }

    // Returns '90210 is a valid zip code'

```

Listing 1: demo/Explore1.java

- *STRING_VAR.split(...)*
  - Splits string by the regex pattern
  - \ W+ means NOT words
  - (...)? means optional, i.e. can be included, but it's okay if its not included



Example:

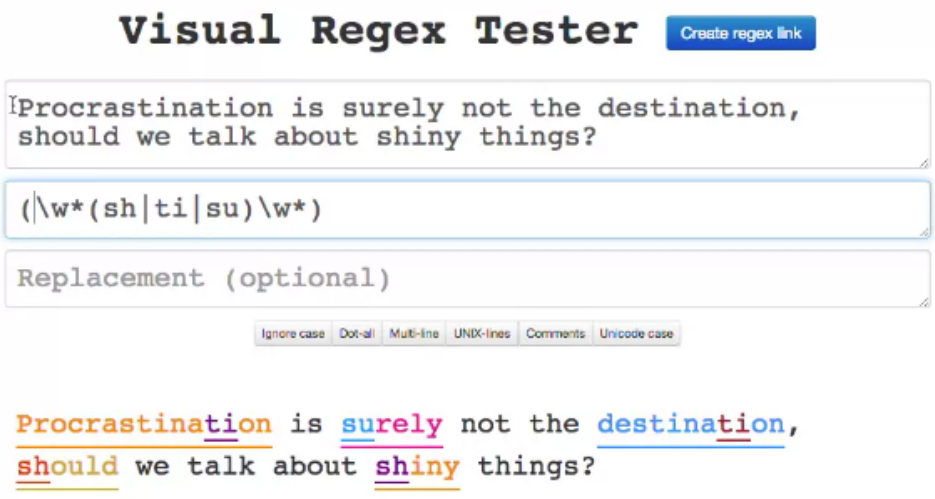
```
    String skills = "JavaScript, HTML, CSS, and Java";
    for (String skill : skills.split("\\W+(and\\W+)?")) {
        System.out.printf("Skill : %s \n", skill);
    }

    // Returns
    // Skill : JavaScript
    // Skill : HTML
    // Skill : CSS
    // Skill : Java

```

Listing 2: demo/Explore2.java

- *Pattern pattern = Pattern.compile(REGEX_PATTERN);*
  Matcher matcher = pattern.matcher(STRING_VAR)
  - Returns all words matching pattern
  - Note: (...) means a group



Example:

```
1     String script = "Procrastination is surely not the destination,
    should we talk about shiny things?";
2
3     Pattern pattern = Pattern.compile("(\\w*(sh|ti|su)\\w*)",
4                                       Pattern.CASE_INSENSITIVE);
5     Matcher matcher = pattern.matcher(script);
6
7     while (matcher.find()) {
8         System.out.printf("%s is a shushy word because of %s \n",
9                           matcher.group(1), // <- returns value of
    outer parenthesis
10                          matcher.group(2)); // <- returns value of
    inner parenthesis
11    }
12
13    // Returns
14    // Procrastination is a shushy word because of ti
15    // surely is a shushy word because of su
16    // destination is a shushy word because of ti
17    // should is a shushy word because of sh
18    // shiny is a shushy word because of sh
19
```

Listing 3: demo/Explore3.java