String Manipulation in R: Fundamentals: Takeaways

Syntax

- `str_sub()`

  enables us to index strings:

  ```
  words <- "Dataquest is awesome"
      > str_sub(words, 1, 9)
      [1] "Dataquest"
  ```

- `str_to_lower()`

  enables us to change all of the letters to lowercase in a given string: ``` str_to_lower(colnames(recent_grads)) [1] "rank" "major_code" "major" "total" "men" [6] "women" "major_category" "sample_size" "employed" "full_time" [11] "part_time" "full_time_year_round" "unemployed" "unemployment_rate" "median" [16] "college_jobs" "non_college_jobs" "low_wage_jobs" ```

- `str_to_upper()`

  enables us to change all of the letters to uppercase in a given string: ``` exclamation <- "hello" str_to_upper(exclamation) [1] "HELLO" ```

- `str_pad()`

  helps with padding strings, while

  `str_trim()`

  lets us trim whitespace or other designated characters from strings: ``` padded_string <- " Dataquest " str_trim(padded_string, side = "both") [1] "Dataquest" str_pad("Dataquest", width = 20, side = "both", pad = " ") [1] " Dataquest " ```

- `str_split()`

  allows us to split a character vector into smaller substrings by splitting on a given character such as a single whitespace: ``` sentence <- "The stringr library is essential to string manipulation." str_split(sentence, " ") [[1]] [1] "The" "stringr" "library" "is" "essential" "to" "string" [8] "manipulation." ```

- `str_c()`

  allows us to concatenate, or combine, strings together:

  ```
  words <- c("String", "concatentation", "via", "function")
      str_c(words, collapse = " ")
      [1] "String concatentation via function"
  ```

- `str_detect()`

allows us to check if a particular substring is contained within a greater string: ``` review <- "I really enjoyed this product, and I thought it was great for the price."
str_detect(review, "great") [1] TRUE ```

- `str_replace()`

lets us exchange the first instance of a given substring with another: ``` review2 <- "I really enjy codnig in R and wnt to lrn more."
str_replace(review2, pattern = "enjy", replacement = "enjoy") [1] "I really enjoy codnig in R and wnt to lrn more." ```

- `str_replace_all()`

lets us exchange all instances of a substring with another: ``` review3 <- "I want to lrn R, and I definitely wnt to lrn more."
str_replace_all(review3, pattern = "lrn", replacement = "learn") [1] "I want to learn R, and I definitely wnt to learn more." ```

Concepts

- Character vectors are also indexed by number, but us must use the `str_sub()` function to index directly on the string. Trying to index the word using square brackets alone will return some unintended results.
- Using regular expression is the act of searching for a particular search pattern within a greater body of text. Regular expression has uses in finding words, string replacement and string removal.

Further Reading

- `stringr` Documentation
- `stringr` 's vignette on regular expression
- More Documentation on regular expression