Strings and Regular Expressions

David Gerard

2019-02-20

Learning Objectives

- Manipulating strings with the stringr package.
- Regular expressions
- Chapter 14 of RDS.
- Work with Strings Cheatsheet.

Strings

- In R, strings (also called "characters") are created and displayed within quotes:

```
x <- "I am a string!"
x
```

```
## [1] "I am a string!"
```

- Anything within quotes is a string, even numbers!

```
y <- "3"
class(y)
```

```
## [1] "character"
```

- You can have a vector of strings.

```
x <- c("I", "am", "a", "string", "vector")
x[2:3]
```

```
## [1] "am" "a"
```

- The backslash `"\"` means that what is after the backslash is special is some way. For example, if you want to put a quotation mark in a string, you can "escape" the quotation mark with a backslash.

```
x <- "As Tolkein said, \"Not all those who wonder are lost\""
writeLines(x)
```

```
## As Tolkein said, "Not all those who wonder are lost"
```

- Above, `writeLines()` will print out the string itself. `print()` will print out the printed representation of the string (with backslashes and all).

```
print(x)
```

```
## [1] "As Tolkein said, \"Not all those who wonder are lost\""
```

- `"\n"` represents a new line.

```
x <- "Not all those\nwho wonder are lost."
writeLines(x)
```

```
## Not all those
## who wonder are lost.
```

- "\t" represents a tab.

```
x <- "Not all those\twho wonder are lost."
writeLines(x)
```

```
## Not all those    who wonder are lost.
```

- You can add any unicode character with a \u followed by the hexidecimal unicode representation of that character.

```
mu <- "\u00b5"
writeLines(mu)
```

```
## µ
```

stringr

- The stringr package contains a lot of convenience functions for manipulating strings (and they are a lot more user friendly than base R's string manipulation functions like `grep()` and `gsub()`).
- stingr is not part of the tidyverse so you have to load it separately.

```
library(tidyverse)
library(stringr)
```