Lecture Assignment 11

Taiki Yamashita

2024-05-09

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.0     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

11.3.5 1)

What are the most important arguments to locale()?

The local object has the arguments to set the following... data and time formats: data_names, data_format, and time_format time zone: tz numbers: decimal_mark, grouping_mark encoding: encoding

11.3.5 2)

What happens if you try and set decimal_mark and grouping_mark to the same character? What happens to the default value of grouping_mark when you set decimal_mark to ",”? What happens to the default value of decimal_mark when you set the grouping_mark to "."?

locale will throw and error if the decimal and grouping marks are set to the same character... locale(decimal_mark = "."", grouping_mark = "."") If the decimal_mark is set to the comma " ,", the grouping mark is set to the period"."...

```
locale(decimal_mark = ",")
```

```
## <locale>
## Numbers:  123.456,78
## Formats:  %AD / %AT
```

```
## Timezone: UTC
## Encoding: UTF-8
## <date_names>
## Days:   Sunday (Sun), Monday (Mon), Tuesday (Tue), Wednesday (Wed), Thursday
##         (Thu), Friday (Fri), Saturday (Sat)
## Months: January (Jan), February (Feb), March (Mar), April (Apr), May (May),
##         June (Jun), July (Jul), August (Aug), September (Sep), October
##         (Oct), November (Nov), December (Dec)
## AM/PM:  AM/PM
```

If the grouping mark is set to a period, the decimal mark is set to a comma...

```
locale(grouping_mark = ".")
```

```
## <locale>
## Numbers:  123.456,78
## Formats:  %AD / %AT
## Timezone: UTC
## Encoding: UTF-8
## <date_names>
## Days:   Sunday (Sun), Monday (Mon), Tuesday (Tue), Wednesday (Wed), Thursday
##         (Thu), Friday (Fri), Saturday (Sat)
## Months: January (Jan), February (Feb), March (Mar), April (Apr), May (May),
##         June (Jun), July (Jul), August (Aug), September (Sep), October
##         (Oct), November (Nov), December (Dec)
## AM/PM:  AM/PM
```

11.3.5 5)

What's the difference between read_csv() and read_csv2()?

The difference between read_csv() and read_csv2() is the delimiter. The read_csv() uses a comma, while the read_csv2() uses a semi-colon. Using the semi-colon is useful when commas are used as the decimal point.

11.3.5 7)

Generate the correct format string to parse each of the following dates and times:

```
d1 <- "January 1, 2010"
d2 <- "2015-Mar-07"
d3 <- "06-Jun-2017"
d4 <- c("August 19 (2015)", "July 1 (2015)")
d5 <- "12/30/14" # Dec 30, 2014
t1 <- "1705"
t2 <- "11:15:10.12 PM"
```

The correct formats are...

```
parse_date(d1, "%B %d, %Y")
```

```
## [1] "2010-01-01"
```

```
parse_date(d2, "%Y-%b-%d")
```

```
## [1] "2015-03-07"
```

```
parse_date(d3, "%d-%b-%Y")
```

```
## [1] "2017-06-06"
```

```
parse_date(d4, "%B %d (%Y)")
```

```
## [1] "2015-08-19" "2015-07-01"
```

```
parse_date(d5, "%m/%d/%y")
```

```
## [1] "2014-12-30"
```

```
parse_time(t1, "%H%M")
```

```
## 17:05:00
```

time t2 uses real seconds...

```
parse_time(t2, "%H:%M:%OS %p")
```

```
## 23:15:10.12
```