Tarea: Procesado de Ficheros y Carga de Datos

Jesus Mudarra Luján

2023-01-09

Pregunta 1

Investiga la documentación para decir cuales son los argumentos más importantes que trae la función `locale()`

- `date_names`: Representaciones en caracteres de los nombres de día y mes. El código de idioma como cadena (pasado a `date_names_lang()`) o un objeto creado por `date_names()`.
- `date_format`, `time_format`: Formatos de fecha y hora por defecto.
- `decimal_mark`, `grouping_mark`: Símbolos utilizados para indicar el decimal, y para trocear números mayores. La marca decimal sólo puede ser `,` o `.`.
- `tz`: Zona horaria por defecto
- `encoding`: Codificación por defecto.
- `asciify`: ¿Deben eliminarse los diacríticos de los nombres de las fechas y convertirse a ASCII? Esto es útil si se trabaja con datos ASCII en los que se ha perdido la ortografía correcta.

Pregunta 2

- Investiga qué ocurre si intentamos configurar a la vez el `decimal_mark` y `grouping_mark` con el mismo carácter.

```
parse_number("9,999.99", locale = locale(grouping_mark = ".", decimal_mark = "."))
```

```
Error: `decimal_mark` and `grouping_mark` must be different
```

Aparece un error indicando que `decimal_mark` y `grouping_mark` deben de ser diferentes.

- ¿Qué valor por defecto toma el `grouping_mark` cuando configuramos el `decimal_mark` al carácter de coma ","?

```
parse_number("9.999,99", locale = locale(decimal_mark = ","))
```

```
[1] 9999.99
```

El valor por defecto que toma `grouping_mark` es el punto ".".

- ¿Qué valor por defecto toma el `decimal_mark` cuando configuramos el `grouping_mark` al carácter de punto "."?

```
parse_number("9.999,99", locale = locale(grouping_mark = "."))
```

```
[1] 9999.99
```

El valor por defecto que toma `decimal_mark` es la coma ",".

Pregunta 3

Investiga qué hace la opción del `locale()` cuando se utiliza junto al `date_format` y al `time_format`. Crea un ejemplo que muestre cuando puede sernos útil.

```
locale(date_format = "%Y-%m-%d", time_format = "%H:%M")
```

```
<locale>
Numbers:  123,456.78
Formats:  %Y-%m-%d / %H:%M
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
```

Pregunta 4

Crea un nuevo objeto locale que encapsule los ajustes más comunes de los parámetros para la carga de los fichero con los que sueles trabajar.

```
locale(
  date_names = "en",
  date_format = "%AD",
  time_format = "%AT",
  decimal_mark = ".",
  grouping_mark = ",",
  tz = "UTC",
  encoding = "UTF-8",
  asciify = FALSE
)
```

```
<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
```

Pregunta 5

Investiga las diferencias entre `read_csv()` y `read_csv2()`?

`read_csv()` utiliza la , como separador y el . como punto decimal. Por otro lado, `read_csv2()` utiliza ; como separador y la , como punto decimal.

Pregunta 6

Investiga las codificaciones que son más frecuentes en Europa y las más comunes en Asia. Usa un poco de Google e iniciativa para investigar acerca de este tema

Las codificaciones más frecuentes en Europa son todas las ISO-8859 o las Windows-125X. Por otra parte, en Asia es más común el uso de Shift JIS o EUC-KR.

Pregunta 7

Genera el formato correcto de string que procesa cada una de las siguientes fechas y horas:

```
v1 <- "May 19, 2018"
v2 <- "2018-May-08"
v3 <- "09-Jul-2013"
v4 <- c("January 19 (2019)", "May 1 (2015)")
v5 <- "12/31/18" # Dic 31, 2014
v6 <- "1305"
v7 <- "12:05:11.15 PM"
```

```
parse_date(v1, format = "%b %d, %Y")
```

```
[1] "2018-05-19"
```

```
parse_date(v2, format = "%Y-%b-%d")
```

```
[1] "2018-05-08"
```

```
parse_date(v3, format = "%d-%b-%Y")
```

```
[1] "2013-07-09"
```

```
parse_date(v4, format = "%B %d (%Y)")
```

```
[1] "2019-01-19" "2015-05-01"
```

```
parse_date(v5, format = "%m/%d/%y")
```

```
[1] "2018-12-31"
```

```
parse_time(v6, format = "%H%M")
```

```
13:05:00
```

```
parse_time(v7, format = "%H:%M:%OS %p")
```

```
12:05:11.15
```