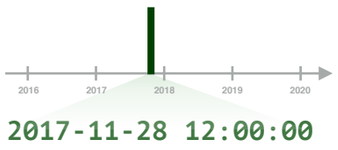
Дата та час з lubridate : : ШПАРГАЛКА

Дата-час



`2017-11-28 12:00:00`
Дата-час (**date-time**) - це точка на осі часу, що зберігається як кількість секунд з моменту 1970-01-01 00:00:00 UTC

*dt <- **as_datetime**(1511870400)*
"2017-11-28 12:00:00 UTC"

`2017-11-28`
Дата (**date**) - це день, що зберігається як к-ть днів з 1970-01-01

*d <- **as_date**(17498)*
"2017-11-28"

`12:00:00`
Об'єкт hms - це **час**, що зберігається як к-ть секунд з 00:00:00

*t <- hms::**as.hms**(85)*
00:01:25

РОЗБОР ДАТИ-ЧАСУ

(Конвертація рядків і чисел в date-time)

1. Визначте в даних порядок наступних елементів: рік (**y**), місяць (**m**), день (**d**), година (**h**), хвилина (**m**) та секунда (**s**).
2. Використовуйте одну з функцій, чиє ім'я відповідає порядку. Кожна приймає багато різних форматів.

`2017-11-28T14:02:00` **ymd_hms**(), **ymd_hm**(), **ymd_h**(). *ymd_hms("2017-11-28T14:02:00")*

`2017-22-12 10:00:00` **ydm_hms**(), **ydm_hm**(), **ydm_h**(). *ydm_hms("2017-22-12 10:00:00")*

`11/28/2017 1:02:03` **mdy_hms**(), **mdy_hm**(), **mdy_h**(). *mdy_hms("11/28/2017 1:02:03")*

`1 Jan 2017 23:59:59` **dmy_hms**(), **dmy_hm**(), **dmy_h**(). *dmy_hms("1 Jan 2017 23:59:59")*

`20170131` **ymd**(), **ydm**(). *ymd(20170131)*

`July 4th, 2000` **mdy**(), **myd**(). *mdy("July 4th, 2000")*

`4th of July '99` **dmy**(), **dym**(). *dmy("4th of July '99")*

`2001: Q3` **yq**() Q - квартал. *yq("2001: Q3")*

`2:01` hms::**hms**() Також lubridate::**hms**(), **hm**() та **ms**(), які повертають періоди.* *hms::hms(sec = 0, min= 1, hours = 2)*

`2017.5` **date_decimal**(decimal, tz = "UTC") *date_decimal(2017.5)*

now(tzone = "") Поточний час у час. поясі (за замовч. - системний). *now()*

today(tzone = "") Поточна дата у час. поясі (за замовч. - системний). *today()*

fast_strptime() Швидкий strptime. *fast_strptime('9/1/01', '%y/%m/%d')*

parse_date_time() Простий strptime. *parse_date_time("9/1/01", "ymd")*

ОТРИМАННЯ І ЗАВДАННЯ КОМПОНЕНТІВ

Використовуйте функції доступу для отримання компонента. *d ## "2017-11-28"*
day(d) ## 28

Присвоюйте результату функції доступу для зміни компонента на місці. *day(d) <- 1*
d ## "2017-11-01"

`2018-01-31 11:59:59` **date**(x) Дата. *date(dt)*

`2018-01-31 11:59:59` **year**(x) Рік. *year(dt)*
isoyear(x) Рік в ISO 8601.
epiyear(x) Епідеміол. рік.

`2018-01-31 11:59:59` **month**(x, label, abbr) Місяць. *month(dt)*

`2018-01-31 11:59:59` **day**(x) День місяця. *day(dt)*
wday(x,label,abbr) День тижня.
qday(x) День квартала.

`2018-01-31 11:59:59` **hour**(x) Години. *hour(dt)*

`2018-01-31 11:59:59` **minute**(x) Хвилини. *minute(dt)*

`2018-01-31 11:59:59` **second**(x) Секунди. *second(dt)*

week(x) Тиждень року. *week(dt)*
isoweek() Тиждень ISO 8601.
epiweek() Епідеміол. тиждень.

quarter(x, with_year = FALSE) Квартал. *quarter(dt)*

semester(x, with_year = FALSE) Півріччя. *semester(dt)*

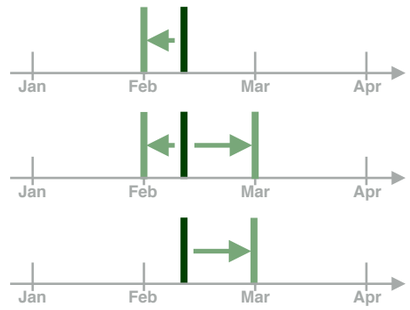
am(x) Перша пол. дня? *am(dt)*
pm(x) Друга пол. дня? *pm(dt)*

dst(x) Літній час? *dst(d)*

leap_year(x) Високосний рік? *leap_year(d)*

update(object, ..., simple = FALSE) *update(dt, mday = 2, hour = 1)*

Округлення



floor_date(x, unit = "second") Округл. вниз до найближ. ел-та. *floor_date(dt, unit = "month")*

round_date(x, unit = "second") Округ. до найближчого ел-та. *round_date(dt, unit = "month")*

ceiling_date(x, unit = "second", change_on_boundary = NULL) Округ. вгору до найближ. ел-та. *ceiling_date(dt, unit = "month")*

rollback(dates, roll_to_first = FALSE, preserve_hms = TRUE) Відкат до останнього дня поперед. місяця. *rollback(dt)*

Шаблони

stamp() Визначає шаблон з еталонного рядка та повертає нову функцію, яка застосовує шаблон до date-time. Також **stamp_date**() та **stamp_time**().

1. Визначення шаблона, створення функції
   sf <- stamp("Created Sunday, Jan 17, 1999 3:34")
2. Застосування шаблона до дат
   sf(ymd("2010-04-05"))
   ## [1] "Created Monday, Apr 05, 2010 00:00"

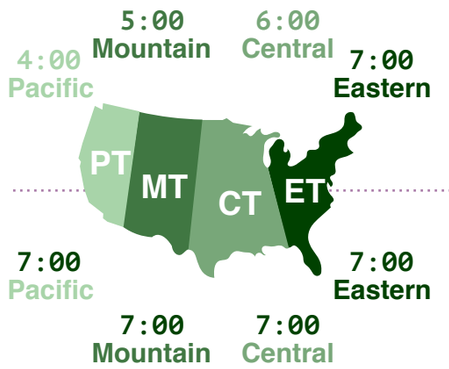
Порада: Вик. дату з day > 12

Часові пояси

R розпізнає ~600 часових поясів. Вони кодують часовий пояс, використання літнього часу та особливості календаря для області. В R використовується один часовий пояс на вектор.

Використовуйте **UTC** для роботи без літнього часу.

OlsonNames() Повертає список доступних часових поясів. *OlsonNames()*



with_tz(time, tzone = "") Повертає **такий же date-time** у новому час. поясі (новий час "на годиннику"). *with_tz(dt, "US/Pacific")*

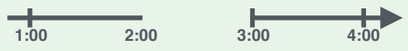
force_tz(time, tzone = "") Повертає **такий же час "на годиннику"** у новому час. поясі (новий date-time). *force_tz(dt, "US/Pacific")*

Операції з датою-часом — В lubridate є три класи для проміжків часу для роботы с датами та date-time

Операції з date-time спираються на **вісь часу**, яка має непостійну поведінку. Приклади:

Нормальний день
nor <- ymd_hms("2018-01-01 01:30:00",tz = "US/Eastern")

Початок літнього часу (година вперед)
gap <- ymd_hms("2018-03-11 01:30:00",tz = "US/Eastern")

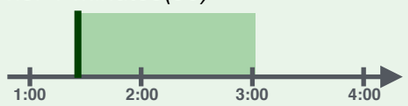


Кінець літнього часу (година назад)
lap <- ymd_hms("2018-11-04 00:30:00",tz = "US/Eastern")

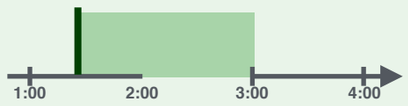
Високосні роки та секунди
leap <- ymd("2019-03-01")

Періоди відображають зміни в часі "на годиннику", ігноруючи особливості осі часу.

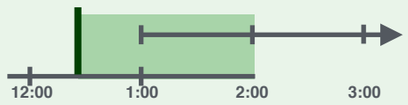
nor + minutes(90)



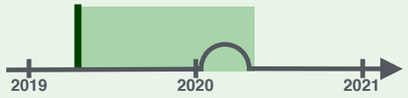
gap + minutes(90)



lap + minutes(90)

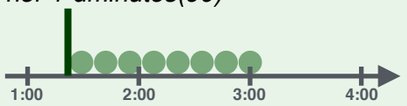


leap + years(1)

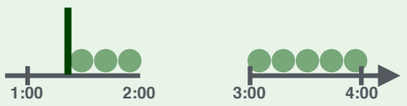


Тривалість відображають плинність фізичного часу, відмінного від часу "на годиннику" в моменти особ-тей осі часу.

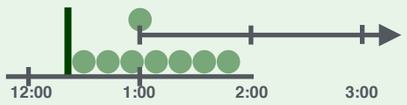
nor + dminutes(90)



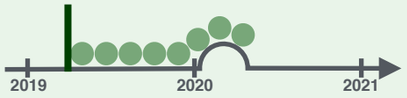
gap + dminutes(90)



lap + dminutes(90)

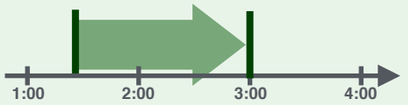


leap + dyears(1)

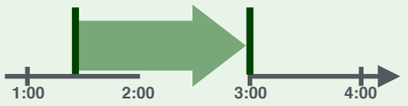


Інтервали представляють певні інтервали осі часу, обмежені початковою та кінц. датою-часом.

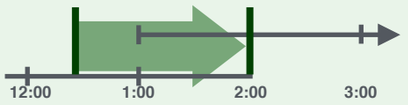
interval(nor, nor + minutes(90))



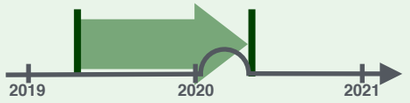
interval(gap, gap + minutes(90))



interval(lap, lap + minutes(90))



interval(leap, leap + years(1))



Не всі роки мають 365 днів через **високос. дні**.

Не всі хвилині мають 60 секунд через **високосні секунди**.

Можливо створити нереальну дату складанням місяців, напр. 31-е лютого

jan31 <- ymd(20180131)
jan31 + months(1)
NA

%m+% та **%m-%** відкочують нереальні дати до крайнього дня попереднього місяця.

jan31 %m+% months(1)
"2018-02-28"

add_with_rollback(e1, e2, roll_to_first = TRUE) відкочують нереальні дати до першого дня нового місяця.

add_with_rollback(jan31, months(1), roll_to_first = TRUE)
"2018-03-01"

ПЕРІОДИ

Додавайте або віднімайте періоди для моделювання подій в певний час "на годиннику", таких як відкриваючий дзвінок NYSE.

Створюйте період за допомогою функції з ім'ям **множини** одиниці виміру, наприклад

p <- months(3) + days(12)
p
"3m 12d 0H 0M 0S"

К-ть місяців | К-ть днів | і т.д.

years(x = 1) x років.
months(x) x місяців.
weeks(x = 1) x тижнів.
days(x = 1) x днів.
hours(x = 1) x годин.
minutes(x = 1) x хвилин.
seconds(x = 1) x секунд.
milliseconds(x = 1) x мілісекунд.
microseconds(x = 1) x мікросекунд.
nanoseconds(x = 1) x наносекунд.
picoseconds(x = 1) x пікосекунд.

period(num = NULL, units = "second", ...) Конструктор періодов, що підходить для автоматизації.
period(5, unit = "years")

as.period(x, unit) Приводить проміжок часу до періоду з можливим завданням одиниці виміру. Також **is.period**(). *as.period(i)*

period_to_seconds(x) Конвертує період в "стандартну" к-ть секунд, заданих періодом. Також **seconds_to_period**().
period_to_seconds(p)

ТРИВАЛІСТЬ

Додавайте або віднімайте тривалість для моделювання фізичних процесів, таких як час роботи батареї. Тривалість зберігається в секундах, єдиній одиниці часу з постійною довжиною. **Difftime** - клас тривалості в базовому R.

Створюйте тривалість за допомогою функції з ім'ям періоду та префіксом ***d***, напр.

dd <- ddays(14)
dd
"1209600s (~2 weeks)"

Точна дов-на в сек. | Еквівалент в звичайних одиницях

dyears(x = 1) 31536000x секунд.
dweeks(x = 1) 604800x секунд.
ddays(x = 1) 86400x секунд.
dhours(x = 1) 3600x секунд.
dminutes(x = 1) 60x секунд.
dseconds(x = 1) x секунд.
dmilliseconds(x = 1) x x 10^{-3} секунд.
dmicroseconds(x = 1) x x 10^{-6} секунд.
dnanoseconds(x = 1) x x 10^{-9} секунд.
dpicoseconds(x = 1) x x 10^{-12} секунд.

duration(num = NULL, units = "second", ...) Конструктор тривалості, що підходить для автоматизації.
duration(5, unit = "years")

as.duration(x, …) Приводить проміжок часу до тривалості. Також **is.duration**(), **is.difftime**(). *as.duration(i)*

make_difftime(x) Створює difftime із заданою кількістю одиниць.
make_difftime(99999)

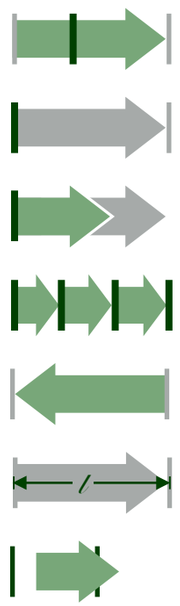
ІНТЕРВАЛИ

Діліть інтервал на тривалість для обчислення його фізичної тривалості, а на період - для обчислення передбачуваної тривалості "на годиннику".

Створюйте інтервал з **interval**() або **%--%**, напр.

Поч. дата | Кін. дата

*i <- **interval**(ymd("2017-01-01"), d)* *## 2017-01-01 UTC--2017-11-28 UTC*
*j <- d **%--%** ymd("2017-12-31")* *## 2017-11-28 UTC--2017-12-31 UTC*



a **%within%** b Потрапляє дата-час *a* в інтервал *b*? *now() %within% i*

int_start(int) Отримання/завдання початкової дати-часу інтервалу. Також **int_end**(). *int_start(i) <- now(); int_start(i)*

int_aligns(int1, int2) Володіють два інтервали спільною границею? Також **int_overlaps**(). *int_aligns(i, j)*

int_diff(times) Створює інтервали між ел-тами date-time вектора.
v <-c(dt, dt + 100, dt + 1000); int_diff(v)

int_flip(int) Змінює напрямок інтервалу. Також **int_standardize**(). *int_flip(i)*

int_length(int) Тривалість в секундах.
int_length(i)

int_shift(int, by) Зміщує інтервал вперед або назад по осі часу на проміжок.
int_shift(i, days(-1))

as.interval(x, start, …) Приводить проміжок часу до інтервалу з початковою датою-часом. Також **is.interval**().
as.interval(days(1), start = now())