



By Rama – own work, CC BY-SA 2.0 fr,
<https://commons.wikimedia.org/w/index.php?curid=481640>

Dates and Times in R

Oleksii Yehorchenkov, Ph.D

Dates and Times in R

R has developed a special representation of dates and times

- Dates are represented by the Date class
- Times are represented by the POSIXct or the POSIXlt class
- Dates are stored internally as the number of days since 1970-01-01
- Times are stored internally as the number of seconds since 1970-01-01

Dates in R

Dates are represented by the Date class and can be coerced from a character string using the `as.Date()` function.

```
x <- as.Date("1970-01-01")
```

```
x
```

```
## [1] "1970-01-01"
```

```
unclass(x)
```

```
## [1] 0
```

```
unclass(as.Date("1970-01-02"))
```

```
## [1] 1
```

Times in R

Times are represented using the POSIXct or the POSIXlt class

- POSIXct is just a very large integer under the hood; it use a useful class when you want to store times in something like a data frame
- POSIXlt is a list underneath and it stores a bunch of other useful information like the day of the week, day of the year, month, day of the month

There are a number of generic functions that work on dates and times

- weekdays: give the day of the week
- months: give the month name
- quarters: give the quarter number (“Q1”, “Q2”, “Q3”, or “Q4”)

Times in R

Times can be coerced from a character string using the `as.POSIXlt` or `as.POSIXct` function.

```
x <- Sys.time()
```

```
x
```

```
## [1] "2017-04-19 19:07:28 EEST"
```

```
p <- as.POSIXlt(x)
```

```
names(unclass(p))
```

```
## [1] "sec" "min" "hour" "mday" "mon"
```

```
## [6] "year" "wday" "yday" "isdst"
```

```
p$sec
```

```
## [1] 28.0137
```

Times in R

You can also use the POSIXct format.

```
x <- Sys.time()
x ## Already in 'POSIXct' format
## [1] "2013-01-24 22:04:14 EST"
unclass(x)
## [1] 1359083054
x$sec
## Error: $ operator is invalid for atomic vectors
p <- as.POSIXlt(x)
p$sec
## [1] 28.0137
```

Times in R

Finally, there is the **strptime** function in case your dates are written in a different format

```
datestring <- c("January 10, 2012 10:40", "December 9, 2011 9:10")
```

```
x <- strptime(datestring, "%B %d, %Y %H:%M")
```

```
x
```

```
## [1] "2012-01-10 10:40:00 EST" "2011-12-09 09:10:00 EST"
```

```
class(x)
```

```
## [1] "POSIXlt" "POSIXt"
```

It is hard to remember the formatting strings. Check `?strptime` for details.

Operations on Dates and Times

You can use mathematical operations on dates and times. Well, really just + and -. You can do comparisons too (i.e. ==, <=)

```
x <- as.Date("2012-01-01")
y <- strptime("9 Jan 2011 11:34:21", "%d %b %Y %H:%M:%S")
x-y
## Warning: Incompatible methods ("-.Date",
## "-.POSIXt") for "-"
## Error: non-numeric argument to binary operator
x <- as.POSIXlt(x)
x-y
## Time difference of 356.3 days
```


Operations on Dates and Times

Even keeps track of leap years, leap seconds, daylight savings, and time zones.

```
x <- as.Date("2012-03-01") y <- as.Date("2012-02-28")
```

```
x-y
```

```
## Time difference of 2 days
```

```
x <- as.POSIXct("2012-10-25 01:00:00")
```

```
y <- as.POSIXct("2012-10-25 06:00:00", tz = "GMT")
```

```
y-x
```

```
## Time difference of 1 hours
```

Summary

- Dates and times have special classes in R that allow for numerical and statistical calculations
- Dates use the Date class
- Times use the POSIXct and POSIXlt class
- Character strings can be coerced to Date/Time classes using the **strptime** function or the **as.Date**, **as.POSIXlt**, or **as.POSIXct**