6. Dates and Times

Contents

1 6. Dates and Times

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

1.1 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") # 0 day from 1970-01-01
x
```

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

```
unclass(x)
```

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

```
unclass(as.Date("1970-01-02")) # one fay from 1970-01-01
```

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

In R, unclass(x) is used to remove the class attribute of an object, x. This basically strips the object of its class information and returns the object as it is stored in its underlying, or base, type.

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

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

```
## [1] "2023-06-25 11:41:41 +03"
```

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

```
## [1] "2023-06-25 11:41:41 +03"
```

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

```
##  [1] "sec"    "min"    "hour"   "mday"   "mon"    "year"   "wday"   "yday"
##  [9] "isdst"  "zone"   "gmtoff"
```

```
p$sec
```

```
## [1] 41.49897
```

You can also use the POSIXct format.

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

```
## [1] "2023-06-25 11:41:41 +03"
```

```
unclass(x)
```

```
## [1] 1687682502
```

```
x$sec
```

```
## Error in x$sec: $ operator is invalid for atomic vectors
```

```
p$sec
```

```
## [1] 41.49897
```

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 EET" "2011-12-09 09:10:00 EET"
```

```
class(x)
```

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

I can never remember the formatting strings. Check ?strptime for details.

```
?strftime
```

1.3 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 in x - y: non-numeric argument to binary operator
```

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

```
## Time difference of 356.6011 days
```

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")
x-y
```

```
## Time difference of -8 hours
```

1.4 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