





Load R Package

- To attach another package to the system you can use the menu or the library function.
- Via the menu: Select the 'Packages' menu and select 'Load package...', a list of available packages on your system will be displayed.
- Select one and click `OK', the package is now attached to your current R session.









Load R Package

- Via the library function:
- The function library can also be used to list all the available libraries on your system with a short description. Run the function without any arguments:
- > library()
- > library(MASS)
- > data()
- >data(phones)









R Workspace

- > getwd() # print the current working directory
- > ls() # list the objects in the current workspace
- > setwd(mydirectory) # change to mydirectory
- > setwd("c:/docs/mydir")









R Workspace

- #view and set options for the session
- > help(options) # learn about availableoptions
- > options() # view current option settings
- > options(digits=3) # number of digits to print on output









R Workspace

- # work with your previous commands
- > history() # display last 25 commands
- history(max.show=Inf) # display all previouscommands
- # save your command history
- > savehistory(file="myfile") # default is ".Rhistory"
- # recall your command history
- > loadhistory(file="myfile") # default is ".Rhistory"









• R programming language provides several functions that deal with date and time. These functions are used to format and convert the date from one form to another form.

Specifier	Description
%a	Abbreviated weekday
%A	Full weekday
%b	Abbreviated month
$\%\mathrm{B}$	Full month
%C	Century









Specifier	Description
%a	Abbreviated weekday
%y	Year without century
$^{0}\!\!/_{\!\!0}\mathrm{Y}$	Year with century
%d	Day of month (01-31)
%j	Day in Year (001-366)
%m	Month of year (01-12)
%D	Data in %m/%d/%y format
%u	Weekday (01-07) Starts on Monday









- # today date
- date<-Sys.Date()
- date
- # abbreviated month
- format(date,format="%a")
- # fullmonth
- format(date,format="%A")
- # weekday
- format(date,format="%u")









- Let's look into the day, month, and year format specifiers to represent dates in different formats.
- # today date
- date<-Sys.Date()</pre>
- # default format yyyy-mm-dd
- date
- # day in month
- format(date,format="%d")
- # month in year
- format(date,format="%m")









- Let's look into the day, month, and year format specifiers to represent dates in different formats.
- # abbreviated month
- format(date,format="%b")
- # full month
- format(date,format="%B")
- # Date
- format(date,format="%D")
- format(date,format="%d-%b-%y")









- # today date
- date<-Sys.Date()
- # year without century
- format(date,format="%y")
- # year with century
- format(date,format="%Y")
- # century
- format(date,format="%C")







Get Date and Time in different Formats in



- # R program to illustrate date function Calling date() function to return current date and time.
- date()
- Sys.Date() function is used to return the system's date.
- # R program to illustrate Sys.Date functio Calling Sys.Date() function to return the system's date
- Sys.Date()
- Sys.timezone() function is used to return the current time zone.
- # R program to illustrate Sys.timezone function Calling Sys.timezone() function to return the current time zone
- Sys.timezone()







Get Date and Time in different Formats in

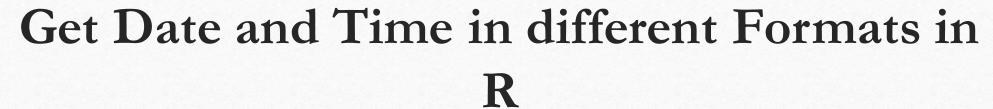


- difftime() function in R Language is used to calculate time difference between dates in the required units like days, weeks, months, years, etc.
- Syntax: difftime(date1, date2, units)
- Parameters:
- date1, date2: Dates to calculate difference
- units: Days, weeks, months, etc.











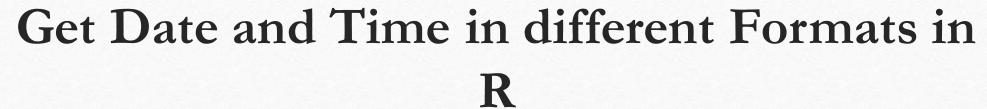
• # R program to find time difference

- # Calling difftime() function
- difftime("2020-5-18", "2020-1-20", units = "days")
- difftime("2020-5-17", "2020-1-18", units = "weeks")











• # R program to find time difference

- # Calling difftime() function
- difftime("2022-5-16", "2020-1-15", units = "hours")
- difftime("2022-5-16", "2020-1-15", units = "mins")









Convert a String into Date Format in R

• as.Date() function in R Language is used to convert a string into date format.

Syntax: as.Date(x, format)

- Parameters:
- x: string variable
- format: Format in which string is declared(\%m/\%d/\%y)









Convert a String into Date Format in R

- # R program to convert string into date
- # Creating a string vector
- dates < c("27/02/92")

•

- # Conversion into date format
- result <- as. Date (dates, " 0 / 0

- # Print result
- print(result)









Convert a String into Date Format in R

- # R program to convert string into date
- # Creating a string vector
- dates <- c("02 / 27 / 92", "02 / 27 / 92",
- "01 / 14 / 92", "02 / 28 / 92",
- **"**02 / 01 / 92**"**)

•

- # Conversion into date format
- result<-as.Date(dates, "% m/% d/% y")

•

- # Print result
- print(result)









How to compare time in R?

```
Method 1: Using logical operators
# declaring a time object
time1 <- as.POSIXct("08:32:07", format = "%H:%M:%S")
print ("Time 1")
print (time1)
time2 <- as.POSIXct("08:32:08", format = "%H:%M:%S")
print ("Time 2")
print (time2)
```









How to compare time in R?

```
if (time1 = time2){
   print("Equal times")
}else{
   if(time1< time2){
       print ("Time1 smaller")
   }else{
       print ("Time2 smaller")
```









Method 2: Using comparison operators

Using difftime() method

Syntax: difftime(time1, time2, tz,units = c("auto", "secs", "mins", "hours", "days", "weeks"))

Parameter:

time1 and time2 – the datetime objects or numeric vectors tz – time zone (optional)

units – the specification of units to perform arithmetic on

Return type: a difftime object applying the arithmetic on datetime object









Method 2: Using comparison operators

```
# declaring a time object
time1 <- as.POSIXct("08:35:07", format = "%H:%M:%S")
print ("Time 1")
print (time1)
time2 <- as.POSIXct("08:32:08", format = "%H:%M:%S")
print ("Time 2")
print (time2)
if (time1 == time2){
    print("Equal times")
```









Method 2: Using comparison operators

```
else{
      # checking if time1 is smaller than time2
      if(time1< time2){
            print ("Time2 - Time1")
            # calculating time2-time1
             difftime(time2,time1, units = "hours")
      }else{
            # calculating time1-time2
            print ("Time1 - Time2")
             difftime(time1,time2, units = "hours")
```









```
0
```

```
# declaring a time object
time1 <- as.POSIXct("08:35:07", format = "%H:%M:%S")
print ("Time 1")
print (time1)
time2 <- as.POSIXct("08:32:08", format = "%H:%M:%S")
print ("Time 2")
print (time2)
if (time1 == time2){
    print("Equal times")
```







Method 3: Using '-' operator

```
0
```

```
# declaring a time object
time1 <- as.POSIXct("09:35:07", format = "%H:%M:%S")
print ("Time 1")
print (time1)
time2 <- as.POSIXct("09:35:08", format = "%H:%M:%S")
print ("Time 2")
print (time2)
```







Method 3: Using '-' operator



```
if (time1 == time2)
       print("Equal times")
}else{
       # checking if time1 is smaller than time2
       if(time1< time2){
              print ("Time2 - Time1")
              # calculating time2-time1
              print (time2 -time1)
       }else{
              # calculating time1-time2
              print ("Time1 - Time2")
              print (time1-time2)
```









THANK YOU

THANK YOU



