r_int_day_2_vector_list_dataframe

Nicholus Tint Zaw

2022-10-27

Content

1. Vector
2. List
3. Data Frame

Vector

Type of vector

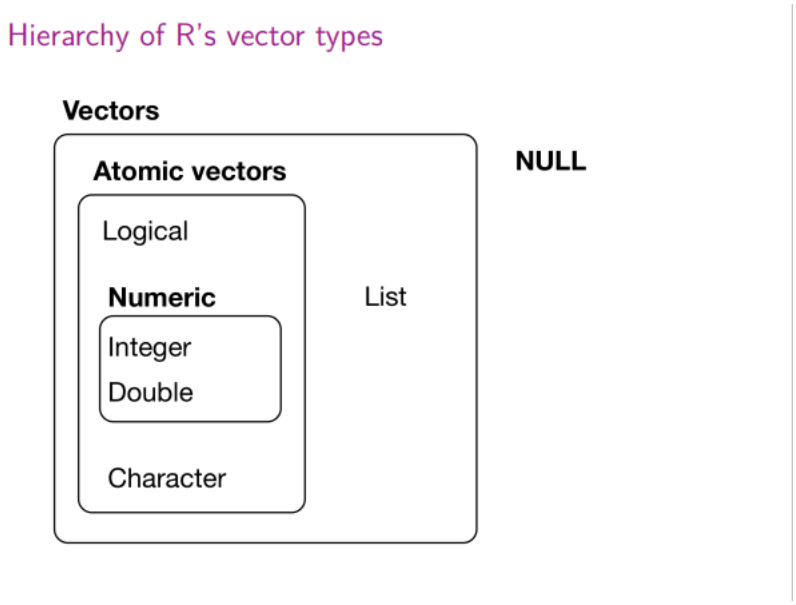

Figure 1: types of vector

```
letters
```

```
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q" "r" "s"
## [20] "t" "u" "v" "w" "x" "y" "z"
```

```r
# type of vector
typeof(letters)
```

```
## [1] "character"
```

```r
typeof(1:10)
```

```
## [1] "integer"
```

```r
# logical

1:7 %% 3 == 0
```

```
## [1] FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE
```

```r
typeof(1:7 %% 3 == 0)
```

```
## [1] "logical"
```

```r
sum(1:7 %% 3 == 0)
```

```
## [1] 2
```

```r
length(1:7 %% 3 == 0)
```

```
## [1] 7
```

```r
# length of vector
length(letters)
```

```
## [1] 26
```

```r
length(1:10)
```

```
## [1] 10
```

numeric: integer vs double

```r
# create a string of double-precision values
dbl_var <- c(1, 2.5, 4.5)
dbl_var
```

```
## [1] 1.0 2.5 4.5
```

```r
# placing an L after the values creates a string of integers
int_var <- c(1L, 6L, 10L)
int_var
```

```
## [1]  1  6 10
```

```r
typeof(dbl_var)
```

```
## [1] "double"
```

```r
typeof(int_var)
```

```
## [1] "integer"
```

```r
# converts integers to double-precision values
as.double(int_var)
```

```
## [1]  1  6 10
```

```r
# identical to as.double()
x <- as.numeric(int_var)
x
```

```
## [1]  1  6 10
```

```r
typeof(x)
```

```
## [1] "double"
```

```r
# converts doubles to integers
as.integer(dbl_var)
```

```
## [1] 1 2 4
```

Implicit coercion: the most complex type wins

```r
typeof(c(TRUE,1))
```

```
## [1] "double"
```

```r
typeof(c(1L,1.5))
```

```
## [1] "double"
```

```r
typeof(c(1.5, "a"))
```

```
## [1] "character"
```

Manipulation of atomic vectors

```r
sample(5) + 100
```

```
## [1] 101 104 102 105 103
```

```r
runif(3) > 0.5
```

```
## [1] FALSE FALSE  TRUE
```

```r
# vector recycling
1:10 + 1:2
```

```
##  [1]  2  4  4  6  6  8  8 10 10 12
```

```r
# naming vector
c("x" = 1, "y" = 5, "z" = 10)
```

```
##  x  y  z
##  1  5 10
```

Filtering vector

```r
# by position
x <- c("one", "two", "three", "four", "five")

x[1]
```

```
## [1] "one"
```

```r
x[c(1,4)]
```

```
## [1] "one"  "four"
```

```r
x[-1]
```

```
## [1] "two"   "three" "four"  "five"
```

```r
x[c(-1, -3)]
```

```
## [1] "two"  "four" "five"
```

```r
# by logical
x <- c(10, 3, NA, 5, 8, 1, NA)

is.na(x)
```

```
## [1] FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE
```

```r
!is.na(x)
```

```
## [1]  TRUE  TRUE FALSE  TRUE  TRUE  TRUE FALSE
```

```r
x[is.na(x)]
```

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

```r
x[!is.na(x)]
```

```
## [1] 10  3  5  8  1
```

```r
x[x %% 2 == 0]
```

```
## [1] 10 NA  8 NA
```

```r
# by character
x <- c("abc" = 1, "def" = 2, "xyz" = 5)

x[c("xyz", "def")]
```

```
## xyz def
##   5   2
```

```r
x %in% "1"
```

```
## [1]  TRUE FALSE FALSE
```

List

```r
x_vec <- c(1,2,3)

x_list <- list(1,2,3)

typeof(x_vec)
```

```
## [1] "double"
```

```
typeof(x_list)
```

```
## [1] "list"
```

```
x_named <- list(a = 1, b = 2, c = 3)
str(x_list)
```

```
## List of 3
##  $ : num 1
##  $ : num 2
##  $ : num 3
```

```
str(x_named)
```

```
## List of 3
##  $ a: num 1
##  $ b: num 2
##  $ c: num 3
```

```
str(x_vec)
```

```
##  num [1:3] 1 2 3
```

how list different from vector

```
y <- list("a", 1L, 1.5, TRUE)
y
```

```
## [[1]]
## [1] "a"
##
## [[2]]
## [1] 1
##
## [[3]]
## [1] 1.5
##
## [[4]]
## [1] TRUE
```

```
list_mix <- list("a", "b", 1:10)
list_mix
```

```
## [[1]]
## [1] "a"
##
## [[2]]
## [1] "b"
##
## [[3]]
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
length(list_mix)
```

```
## [1] 3
```

```
str(list_mix)
```

```
## List of 3
##  $ : chr "a"
##  $ : chr "b"
##  $ : int [1:10] 1 2 3 4 5 6 7 8 9 10
```

```
# nested list
z <- list(list(1,2), list(3,4))
z
```

```
## [[1]]
## [[1]][[1]]
## [1] 1
##
## [[1]][[2]]
## [1] 2
##
##
## [[2]]
## [[2]][[1]]
## [1] 3
##
## [[2]][[2]]
## [1] 4
```

```
str(z)
```

```
## List of 2
##  $ :List of 2
##   ..$ : num 1
##   ..$ : num 2
##  $ :List of 2
##   ..$ : num 3
##   ..$ : num 4
```

```
# Lists convey hierarchical structure
sentences <- "Housed within the Center for International Development (CID), Evidence for Policy Design


sentences
```

```
## [1] "Housed within the Center for International Development (CID), Evidence for Policy Design (EPoD)
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
sentences %>% head(1)
```

```
## [1] "Housed within the Center for International Development (CID), Evidence for Policy Design (EPoD)
```

```
library(stringr)
```

```
##
## Attaching package: 'stringr'

## The following object is masked _by_ '.GlobalEnv':
##
##     sentences
```

```
sentences %>% head(1) %>% str_split(" ")
```

```
## [[1]]
##  [1] "Housed"            "within"            "the"
##  [4] "Center"            "for"               "International"
##  [7] "Development"       "(CID),"            "Evidence"
## [10] "for"               "Policy"            "Design"
## [13] "(EPoD)"            "is"                "a"
## [16] "dynamic"           "research"          "initiative"
## [19] "that"              "brings"            "analytical"
## [22] "insights,"         "typically"         "from"
## [25] "economics,"        "to"                "the"
## [28] "design"            "and"               "implementation"
## [31] "of"                "public"            "policies"
## [34] "and"               "programs"          "around"
## [37] "the"               "world."            "EPoD"
## [40] "directly"          "engages"           "with"
## [43] "governments"       "and"               "local"
## [46] "organizations"     "to"                "identify"
## [49] "key"               "questions,"        "design"
## [52] "innovative"        "new"               "policies"
## [55] "or"                "interventions,"    "and"
## [58] "test"              "these"             "using"
## [61] "the"               "tools"             "of"
```

```
## [64] "applied"           "microeconomics,"   "including"
## [67] "large"             "field-based"       "experiments."
## [70] "Current"           "research"          "topics"
## [73] "at"                "EPoD"              "include"
## [76] "governance,"       "social"            "protection,"
## [79] "education,"        "entrepreneurship," "health,"
## [82] "skills,"           "state"             "capacity,"
## [85] "sustainable"       "development,"      "and"
## [88] "access"            "to"                "finance."
## [91] ""
```

Subsetting list

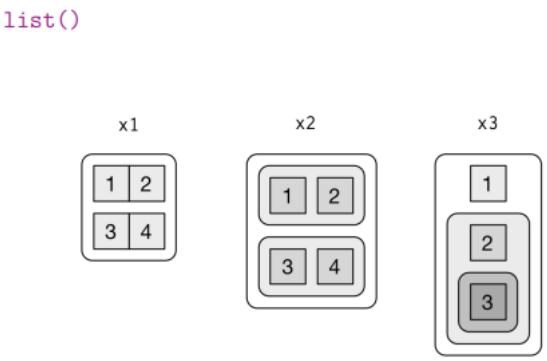

Figure 2: Diagram of different list structure

code	concept
x	shaker
x[1]	shaker with a packet
x[2]	shaker with a different packet
x[1:2]	shaker with two packets
x[[1]]	packet
x[[1]][[1]]	content of packet

Figure 3: code vs concept

Code demostration

```
a <- list(a = 1:3,
          b = "a string",
          c = pi,
```

```
          d = list(-1, -5))
str(a)
```

```
## List of 4
##  $ a: int [1:3] 1 2 3
##  $ b: chr "a string"
##  $ c: num 3.14
##  $ d:List of 2
##   ..$ : num -1
##   ..$ : num -5
```

```
a
```

```
## $a
## [1] 1 2 3
##
## $b
## [1] "a string"
##
## $c
## [1] 3.141593
##
## $d
## $d[[1]]
## [1] -1
##
## $d[[2]]
## [1] -5
```

```
a[1]
```

```
## $a
## [1] 1 2 3
```

```
a[c(1,3)]
```

```
## $a
## [1] 1 2 3
##
## $c
## [1] 3.141593
```

```
a[[1]]
```

```
## [1] 1 2 3
```

```
typeof(a)
```

```
## [1] "list"
```

```
typeof(a[1])
```

```
## [1] "list"
```

```
typeof(a[[1]])
```

```
## [1] "integer"
```

```
a[[1]][[1]]
```

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

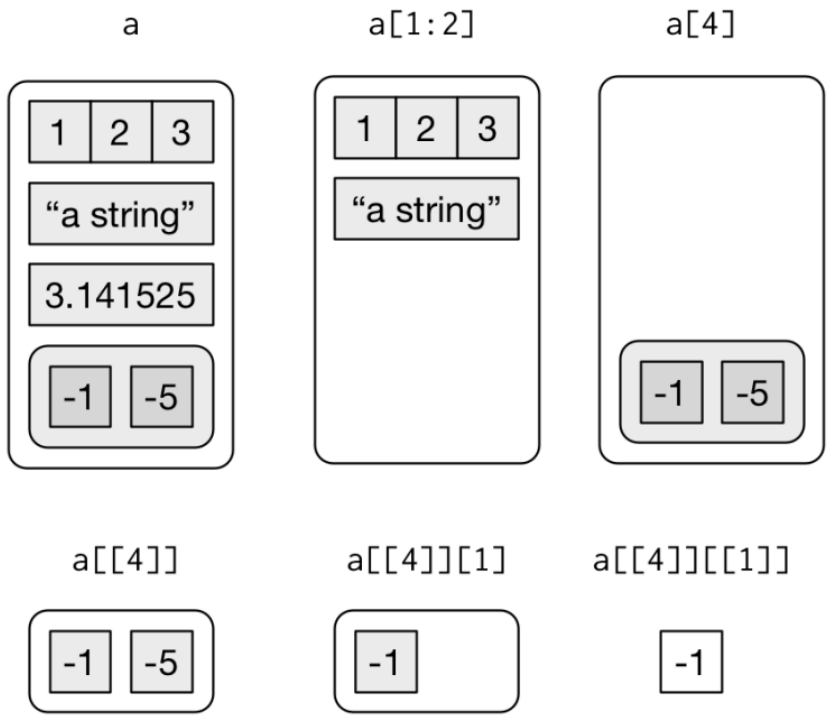

Figure 4: Visual demostration

Data Frame

```
x_tibble <- dplyr::tibble("x" = 1:10, "y" = 21:30)
class(x_tibble)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
as.data.frame(x_tibble)
```

```
##     x  y
## 1   1 21
## 2   2 22
## 3   3 23
## 4   4 24
## 5   5 25
## 6   6 26
## 7   7 27
## 8   8 28
## 9   9 29
## 10 10 30
```

```
as.matrix(x_tibble)
```

```
##        x  y
##  [1,]  1 21
##  [2,]  2 22
##  [3,]  3 23
##  [4,]  4 24
##  [5,]  5 25
##  [6,]  6 26
##  [7,]  7 27
##  [8,]  8 28
##  [9,]  9 29
## [10,] 10 30
```

```
# Create the data frame
df <-   data.frame(
   emp_id = c (6:8),
   emp_name = c("Rasmi","Pranab","Tusar"),
   salary = c(578.0,722.5,632.8),
   start_date = as.Date(c("2013-05-21","2013-07-30","2014-06-17")),
   dept = c("IT","Operations","Fianance"),
   stringsAsFactors = FALSE
)

df
```

```
##   emp_id emp_name salary start_date       dept
## 1      6    Rasmi  578.0 2013-05-21         IT
## 2      7   Pranab  722.5 2013-07-30 Operations
## 3      8    Tusar  632.8 2014-06-17   Fianance
```

```
class(df)
```

```
## [1] "data.frame"
```