

Advanced R in Korean

Jsang

2020-01-10

Contents

Preface	1
1 Introduction	3
2 Names and values	5
2.1 Introduction	5
2.2 Binding basics	6
2.3 Copy-on-modify	8
2.4 Object size	14
2.5 Modify-in-place	16
2.6 Unbinding and the garbage collector	20
2.7 Summary	20

Preface

- Wickham Advanced R .
- .
- .
- , / .
- (@) .

Chapter 1

Introduction

Chapter 2

Names and values

2.1 Introduction

R (object) (name) . ,

-
-
- R

(names) (values) , R (object) .

Quiz

. Section 2.7 .

1. , 1 2 “3” ? [[, \$. 1, 2, 3
?

```
df <- data.frame(runif(3), runif(3))
names(df) <- c(1, 2)
```

2. , y ?

```
x <- runif(1e6)
y <- list(x, x, x)
```

3. a ?

```
a <- c(1, 5, 3, 2)
b <- a
b[[1]] <- 10
```

Outline

- 2.2 (names) (values) , <- (binding) (reference)
- 2.3 R ' ' , . tracemem()
- 2.4 (object) , . ,
utils::object.size() , lobstr::obj_size()
- 2.5 ‘copy-on-modify’(@ , place .(@))
(environments) ,
- 2.6 , garbage collector

Prerequisites

R lobstr .

```
library(lobstr)
```

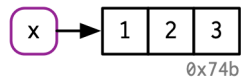
2.2 Binding basics

```
x <- c(1, 2, 3)
```

"x , 1, 2, 3 . " , R

- c(1, 2, 3) .
- x .

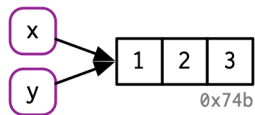
, .
, .



```

x      ,      .      c(1, 2, 3)      (      ,      )      .
  <-      , <-      (binding)      .
      , (name) (value) (reference)      .(@      '      ?)
      ,      c(1, 2, 3)      .      '      .(@      ,
      )
  
```

```
y <- x
```



```

c(1, 2, 3) 0x74b
  (ID)      ID      , ,      .      .      .
      , ID      .
lobstr::obj_addr()      ID      .      x y      ID      .
  
```

```

obj_addr(x)
#> [1] "0x18308880"
obj_addr(y)
#> [1] "0x18308880"
  
```

```

ID      R      .
      .      ,      .      .
  
```

2.2.1 Non-syntactic names

```

R      .      (syntactic)      , , . , -      , -      .
  TRUE, NULL, if,      function      (reserved words)      .(@      R      .
      )      non-syntactic      ,      .
  
```

```

_abc <- 1
#> Error: unexpected input in "_"

if <- 10
#> Error: unexpected assignment in "if <-"
  
```

backticks

```
`_abc` <- 1
`_abc`
#> [1] 1
`if` <- 10
`if`
#> [1] 10
```

, R

2.2.2 Exercises

1. a, b, c, d

```
a <- 1:10
b <- a
c <- b
d <- 1:10
```

2. `? lobster::obj_addr()`

```
mean
base::mean
get("mean")
evalq(mean)
match.fun("mean")
```

3. `read.csv` R, non-syntactic syntactic
?
4. non-syntactic syntactic `make.names()` ?
5. syntactic `.123e1` syntactic ?
`?makes.names`

2.3 Copy-on-modify

`x y` `y`


```
y[[3]] <- 5L
```

```
untracemem(x)
```

```
untracemem() tracemem() , .
```

2.3.2 Function calls

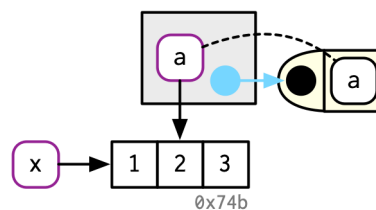
```
f <- function(a) {
  a
}

x <- c(1, 2, 3)
cat(tracemem(x), "\n")
#> <000000001A2688D0>

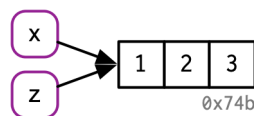
z <- f(x)
# there's no copy here!

untracemem(x)
```

```
f() , a x .
```



7.4.4 . , f() . argument a
 , () () .
 f() , x z . 0x74b , . f() x , R
 , z .

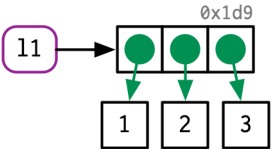


2.3.3 Lists

(,) . , .

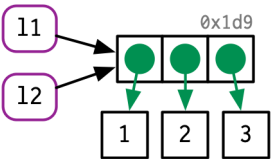
```
11 <- list(1, 2, 3)
```

, , () .

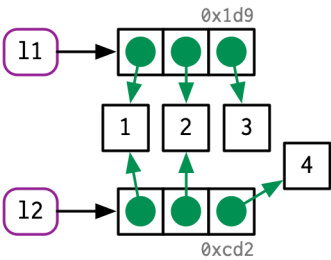


.

```
12 <- 11
```



```
12[[3]] <- 4
```



, copy-on-modify . , . R 3.1.0
, , .
, lobstr::ref() . ref() ID ,

```

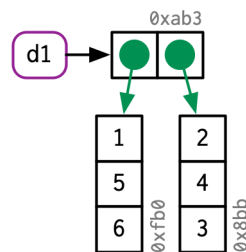
ref(11, 12)
#> o [1:0x17fa8250] <list>
#> +- [2:0x17f958d8] <dbl>
#> +- [3:0x17f958a0] <dbl>
#> \- [4:0x17f95868] <dbl>
#>
#> o [5:0x18843c58] <list>
#> +- [2:0x17f958d8]
#> +- [3:0x17f958a0]
#> \- [6:0x18a12310] <dbl>

```

2.3.4 Data frames

copy-on-modify

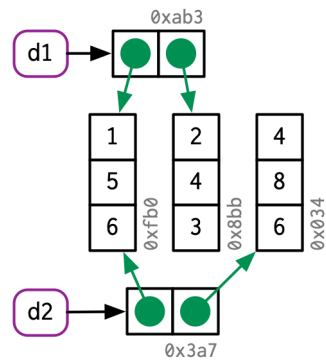
```
d1 <- data.frame(x = c(1, 5, 6), y = c(2, 4, 3))
```



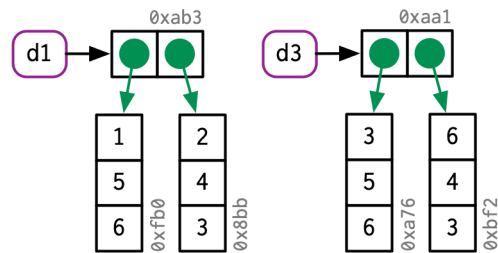
```

d2 <- d1
d2[, 2] <- d2[, 2] * 2

```



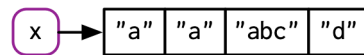

```
d3 <- d1
d3[1, ] <- d3[1, ] * 3
```



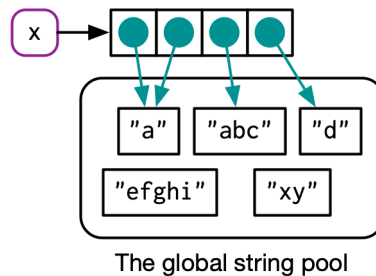
2.3.5 Character vectors

R

```
x <- c("a", "a", "abc", "d")
```



R global string pool



```
ref() character TRUE
```

```
ref(x, character = TRUE)
#> 0 [1:0x18fd9000] <chr>
#> +-[2:0x12f376d0] <string: "a">
#> +-[2:0x12f376d0]
#> +-[3:0x1994f480] <string: "abc">
#> \-[4:0x13135330] <string: "d">
```

2.3.6 Exercises

1. `tracemem(1:10)` ?
2. , `tracemem()` . :

```
x <- c(1L, 2L, 3L)
tracemem(x)

x[[3]] <- 4
```

3. .

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

4. ?

```
x <- list(1:10)
x[[2]] <- x
```

2.4 Object size

```
lobstr::obj_size()
```

```
obj_size(letters)
#> 1,712 B
obj_size(ggplot2::diamonds)
#> 3,456,344 B
```

```
x <- runif(1e6)
obj_size(x)
#> 8,000,048 B

y <- list(x, x, x)
obj_size(y)
#> 8,000,128 B
```

`y` `x` 80 , .

```
obj_size(list(NULL, NULL, NULL))
#> 80 B
```

, R global string pool . 100 100 .

```
banana <- "bananas bananas bananas"
obj_size(banana)
#> 136 B
obj_size(rep(banana, 100))
#> 928 B
```

`y` . , `x` `y` `y` . `obj_size(x) + obj_size(y)` `obj_size(x,`

```
obj_size(x, y)
#> 8,000,128 B
```

, 3.5.0 R ALTREP , alternative representation . R . : , .

```
obj_size(1:3)
#> 680 B
obj_size(1:1e3)
#> 680 B
obj_size(1:1e6)
#> 680 B
obj_size(1:1e9)
#> 680 B
```

2.4.1 Exercises

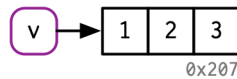
1. , `object.size(y)` `obj_size(y)` ? `object.size()` .

```
y <- rep(list(runif(1e4)), 100)

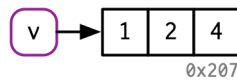
object.size(y)
#> 8005648 bytes
obj_size(y)
#> 80,896 B
```



```
v <- c(1, 2, 3)
```



```
v[[3]] <- 4
```



```
( ID , v 0bx207 .)
R .
```

- , R 0, 1 . , , 1
- , . “ ” C . R-core

```
. R For , .
```

```
x <- data.frame(matrix(runif(5 * 1e4), ncol = 5))
medians <- vapply(x, median, numeric(1))

for (i in seq_along(medians)) {
  x[[i]] <- x[[i]] - medians[[i]]
}
```

```
loop , loop . tracemem()
```

```
cat(tracemem(x), "\n")

#> <0x7f80c429e020>
for (i in 1:5) {
  x[[i]] <- x[[i]] - medians[[i]]
}

#> tracemem[0x7f80c429e020 -> 0x7f80c0c144d8]:
#> tracemem[0x7f80c0c144d8 -> 0x7f80c0c14540]: [[<-data.frame [[<-
#> tracemem[0x7f80c0c14540 -> 0x7f80c0c145a8]: [[<-data.frame [[<-
```

```
#> tracemem[0x7f80c0c145a8 -> 0x7f80c0c14610]:
#> tracemem[0x7f80c0c14610 -> 0x7f80c0c14678]: [[<-.data.frame [[<-
#> tracemem[0x7f80c0c14678 -> 0x7f80c0c146e0]: [[<-.data.frame [[<-
#> tracemem[0x7f80c0c146e0 -> 0x7f80c0c14748]:
#> tracemem[0x7f80c0c14748 -> 0x7f80c0c147b0]: [[<-.data.frame [[<-
#> tracemem[0x7f80c0c147b0 -> 0x7f80c0c14818]: [[<-.data.frame [[<-
#> tracemem[0x7f80c0c14818 -> 0x7f80c0c14880]:
#> tracemem[0x7f80c0c14880 -> 0x7f80c0c148e8]: [[<-.data.frame [[<-
#> tracemem[0x7f80c0c148e8 -> 0x7f80c0c14950]: [[<-.data.frame [[<-
#> tracemem[0x7f80c0c14950 -> 0x7f80c0c149b8]:
#> tracemem[0x7f80c0c149b8 -> 0x7f80c0c14a20]: [[<-.data.frame [[<-
#> tracemem[0x7f80c0c14a20 -> 0x7f80c0c14a88]: [[<-.data.frame [[<-

untracemem(x)
```

,
[[.data.frame x , ,
(regular function) ,.(@ ...)
C , .

```
y <- as.list(x)
cat(tracemem(y), "\n")
#> <0x7f80c5c3de20>

for (i in 1:5) {
  y[[i]] <- y[[i]] - medians[[i]]
}
#> tracemem[0x7f80c5c3de20 -> 0x7f80c48de210]:
```

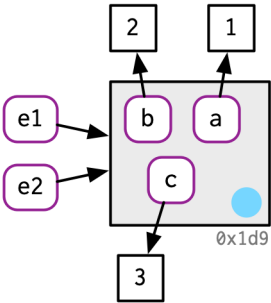
, . , 25 C++

2.5.2 Environments

7 , . (modified
in place) reference semantics ,

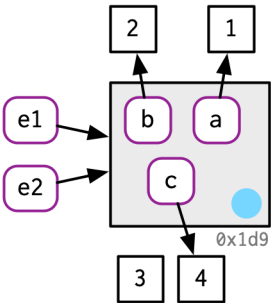
. e1 e2 .

```
e1 <- rlang::env(a = 1, b = 2, c = 3)
e2 <- e1
```



, modified in place .

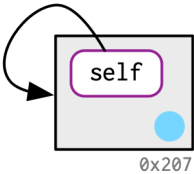
```
e1$c <- 4
e2$c
#> [1] 4
```



“ ” . 10.2.4 . 14 , R6

```
e <- rlang::env()
e$self <- e

ref(e)
#> o [1:0x18631cf8] <env>
#> \-self = [1:0x18631cf8]
```



!

2.5.3 Exercises

1. .

```
x <- list()
x[[1]] <- x
```

2. , 'bench' . ?

3. tracemem() ?

2.6 Unbinding and the garbage collector

2.7 Summary