CHAPTER 14. Pipes with magrittr

PART III. Program

programming skills 을 향상

Chapter 14 : pipe, %>%

Chapter 15 : 함수

Chapter 16 : vector

Chapter 17 : Iteration with purr

Prerequisites

```
library(magrittr)

## Warning: package 'magrittr' was built under R version 3.4.4
```

Piping Alternatives

손동작으로 표현하는 어린이 동요

```
Little bunny Foo Foo
Went hopping through the forest
Scooping up the field mice
And bopping them on the head
```

little bunny Foo Foo 표현하면, foo_foo <- little_bunny()

동작은 hop(), scoop(), bop() 함수로 표현.

4 가지로 코드화 할 수 있음. - Save each intermediate step as a new object. - Overwrite the original object many times. - Compose functions. - Use the pipe

Intermediate Steps

```
foo_foo_1 <- hop(foo_foo, through = forest)
foo_foo_2 <- scoop(foo_foo_1, up = field_mice)
foo_foo_3 <- bop(foo_foo_2, on = head)
```

이 코드의 문제점 - The code is cluttered with unimportant names. - You have to carefully increment the suffix on each line. - 변수 이름을 정확히 입력 필요.

메모리에 중복 저장은 발생하지 않음.(dataframe 간 컬럼 공유)

ggplot2::diamonds 에 새 열을 추가하는 실제 데이터의 용량을 확인

```
if(!(require(pryr))) install.packages("pryr")

## Loading required package: pryr

## Warning: package 'pryr' was built under R version 3.4.4

library(pryr)

diamonds <- ggplot2::diamonds
diamonds2 <- diamonds %>%
    dplyr::mutate(price_per_carat = price / carat)

## Warning: package 'bindrcpp' was built under R version 3.4.4

pryr::object_size(diamonds)

## 3.46 MB

pryr::object_size(diamonds2)

## 3.89 MB

pryr::object_size(diamonds, diamonds2)

## 3.89 MB
```

- diamonds takes up 3.46 MB.
- diamonds2 takes up 3.89 MB.
- diamonds and diamonds2 together take up 3.89 MB!

dataframe 간 값이 변경되지 않으면, 공유하고 변경되면 데이터를 카피

```
diamonds$carat[1] <- NA
pryr::object_size(diamonds)

## 3.46 MB

pryr::object_size(diamonds2)

## 3.89 MB

pryr::object_size(diamonds, diamonds2)
```

```
## 4.32 MB
```

Overwrite the Original

```
foo_foo <- hop(foo_foo, through = forest)
foo_foo <- scoop(foo_foo, up = field_mice)
foo_foo <- bop(foo_foo, on = head)
```

two problems - 실수를했다면 처음부터 전체 파이프 라인을 재실행 필요. - foo_foo 변수 이름을 변경하려면 6 번 수정이 필요.

Function Composition

알아보기 힘들고 개발도 힘듬.

```
bop(
    scoop(
        hop(foo_foo, through = forest),
        up = field_mice
    ),
    on = head
)
```

Use the Pipe

장점은 ??

```
foo_foo %>%
    hop(through = forest) %>%
    scoop(up = field_mouse) %>%
    bop(on = head)
```

magrittr 패키지 내부에서는 아래와 같이 동작

```
my_pipe <- function(.) {
   . <- hop(., through = forest)
   . <- scoop(., up = field_mice)
   bop(., on = head)
}
my_pipe(foo_foo)
```

pipe 가 잘 동작하지 않는 함수 종류 2 가지 - Functions that use the current environment.

```
assign("x", 10)
x

## [1] 10
```

100 이란 값이 설정 안됨.

```
"x" %>% assign(100)
x

## [1] 10
```

환경변수를 넘겨주면 가능함. 이렇게 할려면 안 하고 만다.

```
env <- environment()
"x" %>% assign(100, envir = env)
x

## [1] 100
```

- Functions that use lazy evaluation.

```
tryCatch(stop("!"), error = function(e) "An error")

## [1] "An error"

#stop("!") %>%
#   tryCatch(error = function(e) "An error")
```

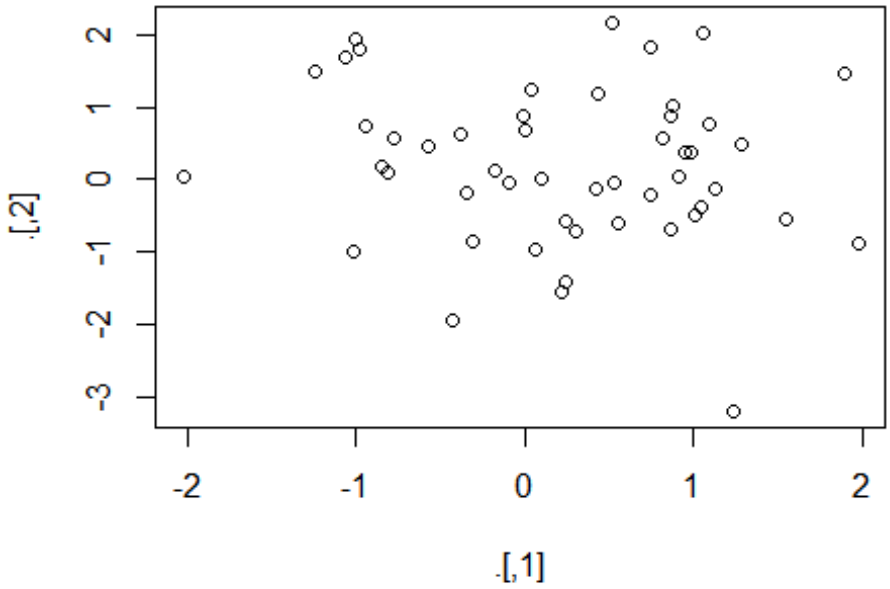
When Not to Use the Pipe

pipe is a powerful tool 이지만, 만능이 아님.

다른 도구를 고려해야할때. - longer than (say) 10 steps. => 중간결과를 디버깅이 필요해져서 intermediate objects 가 유의미한 이름으로 저장이 필요 - multiple inputs or outputs - 작업이 복잡한 그래프 구조를 갖었을때.

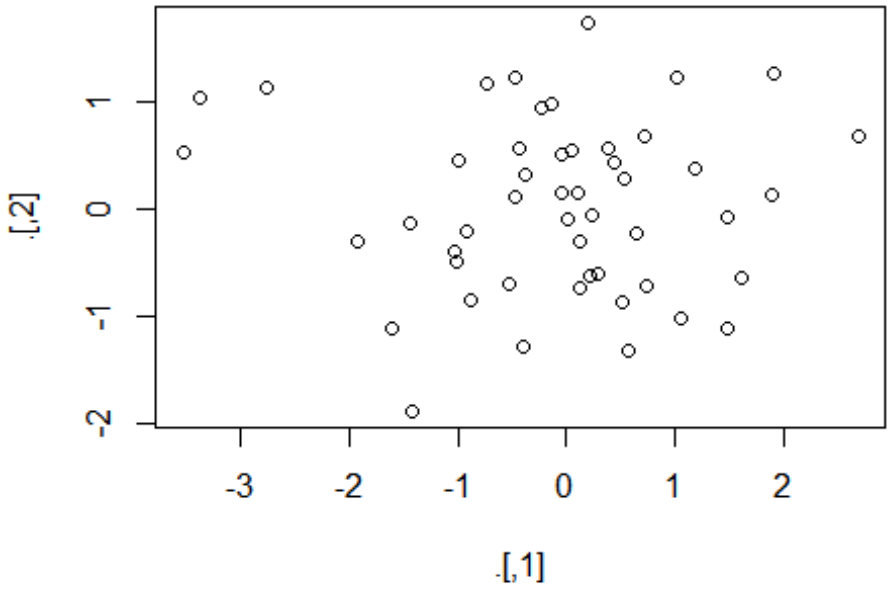
“tee” pipe. %T>% : 오른쪽 반환 대신에 왼쪽 반환

```
rnorm(100) %>%
    matrix(ncol = 2) %>%
    plot() %>%
    str()
```




```
##  NULL

rnorm(100) %>%
    matrix(ncol = 2) %T>%
    plot() %>%
    str()
```




```
##  num [1:50, 1:2] 0.132 1.898 0.433 0.64 0.716 ...
```

%$% 은 dataframe 에서 사용할 수 없는 함수를 사용할때 유용.

```
mtcars %$%
   cor(disp, mpg)

## [1] -0.8475514

#mtcars %>%
#   cor(disp, mpg)
```

%<>% 아래 코드를 단순하게 처리가 가능함.

```
mtcars <- mtcars %>%
    transform(cyl = cyl * 2)

mtcars %>%
    transform(cyl = cyl * 2)

##                    mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0  24 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0  24 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8  16 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4  24 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7  32 360.0 175 3.15 3.440 17.02  0  0    3    2
```

```
## Valiant             18.1  24 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360          14.3  32 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D           24.4  16 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230            22.8  16 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280            19.2  24 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C           17.8  24 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE          16.4  32 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          17.3  32 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC         15.2  32 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4  32 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4  32 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7  32 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            32.4  16  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         30.4  16  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      33.9  16  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5  16 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5  32 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2  32 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3  32 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2  32 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3  16  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0  16 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4  16  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8  32 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7  24 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0  32 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4  16 121.0 109 4.11 2.780 18.60  1  1    4    2
```