R language and data analysis: debug

Qiang Shen

Jan.9,2018

debug

"Finding your bug is a process of confirming the many things that you believe are true — until you find one which is not true."

—Norm Matloff

debug

- message
- warning
- error

debug

```
library(dplyr)
detach("package:dplyr")
suppressMessages(library(dplyr))

log(-5)  #warning
myfunc=function(x) {
  if(x>0) print(x)
    else print(-x)
}
myfunc(3);myfunc(-3)
x=log(-2);myfunc(x) #error
```

debug

- traceback()
- debug()/debugonce()
- browser()/breakpoint

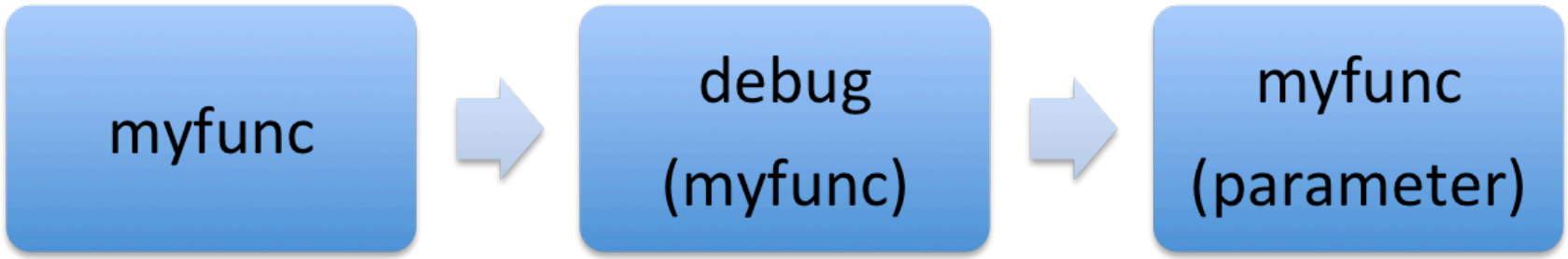
traceback()

traceback()

```
f <- function(x, y){
  z <- x + y
  g(z)
}
g <- function(x){
  z <- round(x)
  h(z)
}

h <- function(x){
  set.seed(1234)
  z <- rnorm(x)
  print(z)
}
f(2,3)
f(2, -3)
traceback()
```

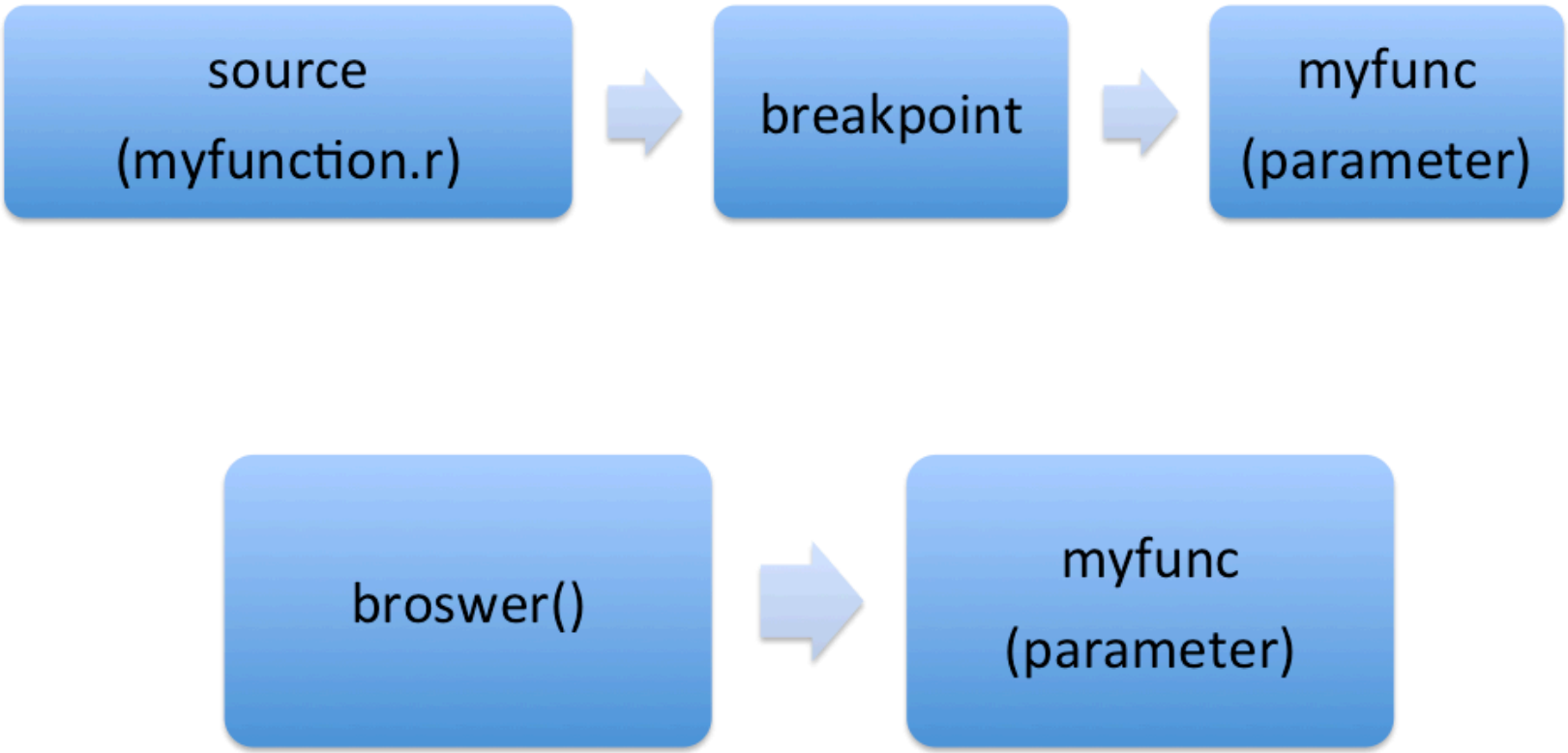
debug()/debugonce()



debug()/debugonce()

```
std<-function(x){
n <- length(x)
meanx <- sum(x)/n
css <- sum((x-meanx)**2)
df=n
sdx <- sqrt(css/df)
return(sdx)
}
x=1:10
sd(x)
std(1:10)
debug(std) #n(next),where,Q(quit)
options(error = browser())
undebug(std)
##debugonce()
##debugSource():Rstudio
```

browser() / breakpoints



browser() / breakpoints

```
std<-function(x){
n <- length(x)
meanx <- sum(x)/n
css <- sum((x-meanx)**2)
df=n
browser()
sdx <- sqrt(css/df)
return(sdx)
}
source('std.r')
x=1:10
std(x)
```

recover

```
f <- function(x, y){
  z <- x + y
  g(z)
}
g <- function(x){
  z <- round(x)
  h(z)
}

h <- function(x){
  set.seed(1234)
  z <- rnorm(x)
  print(z)
}

f(2,3)
f(2, -3)
```

debug summary

- traceback()
- debug()/debugonce()
- browser()/breakpoint
- recover options (optional)