Introduction to Functions

INTERMEDIATE R

Filip Schouwenaars
DataCamp Instructor

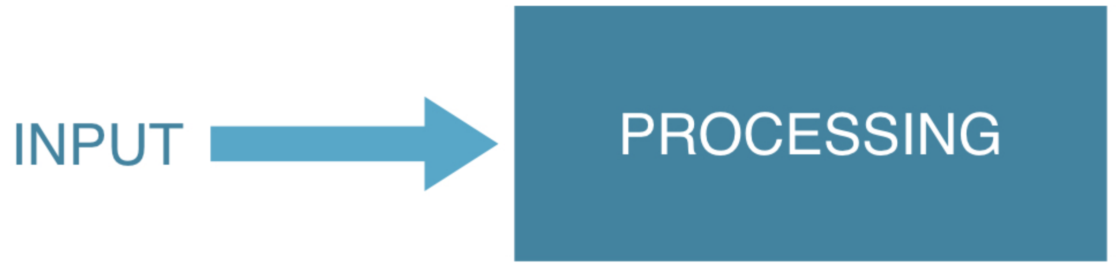
Functions

- You already know 'em!
- Create a list: `list()`
- Display a variable: `print()`

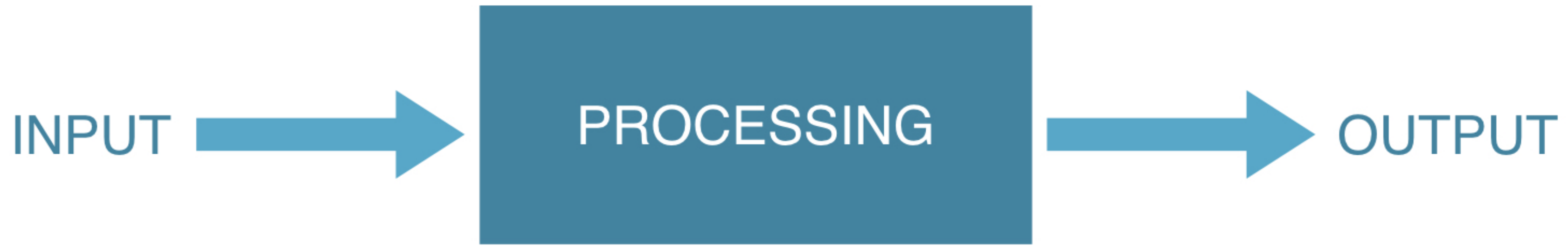
Black box principle

Black box principle

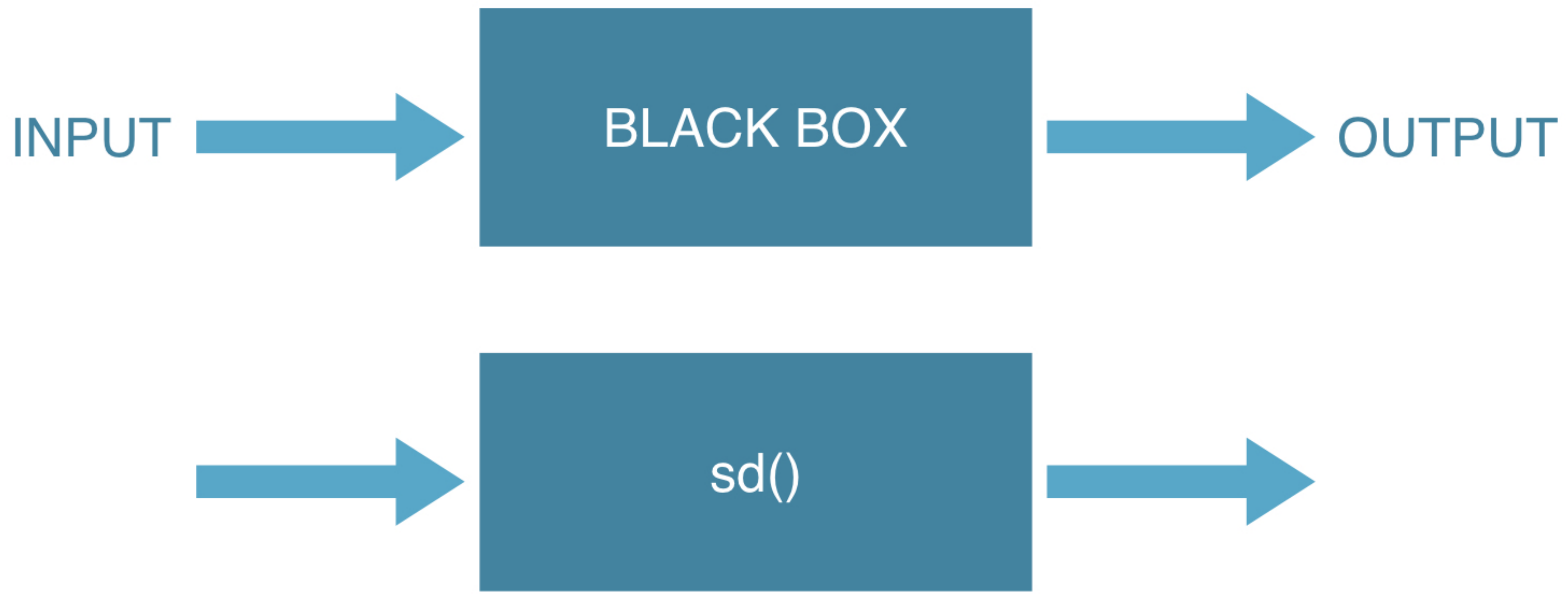
Black box principle



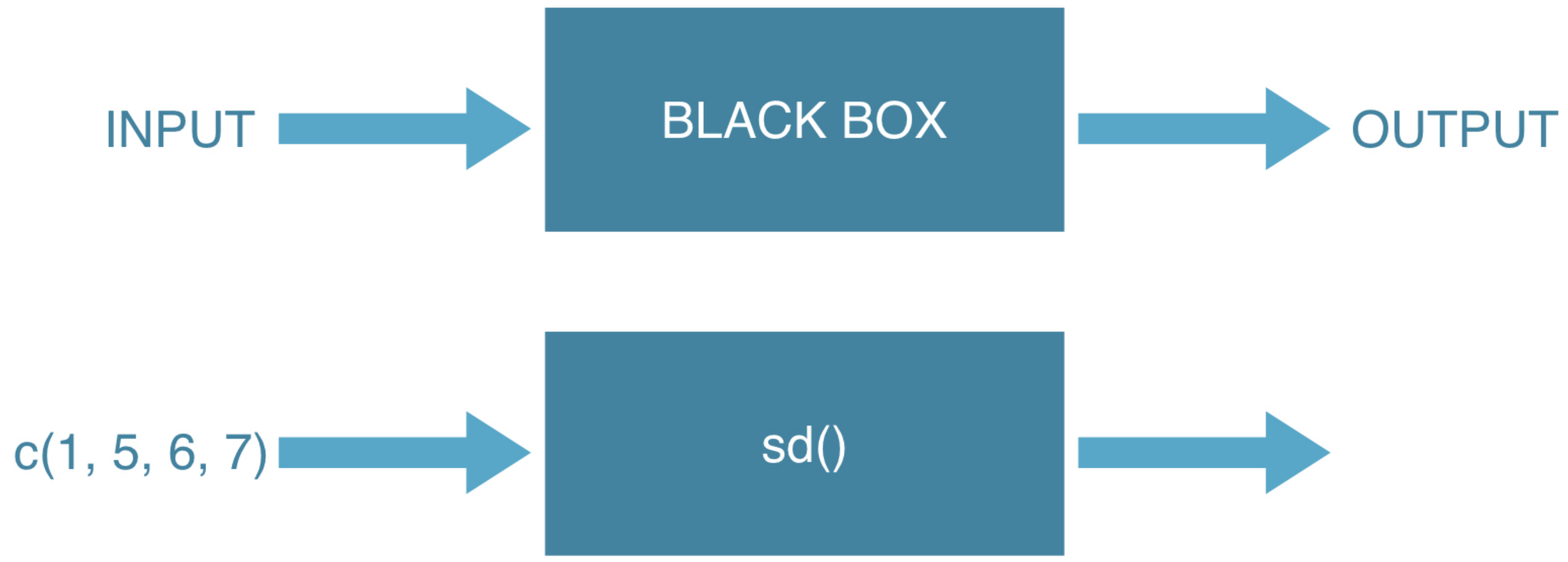
Black box principle



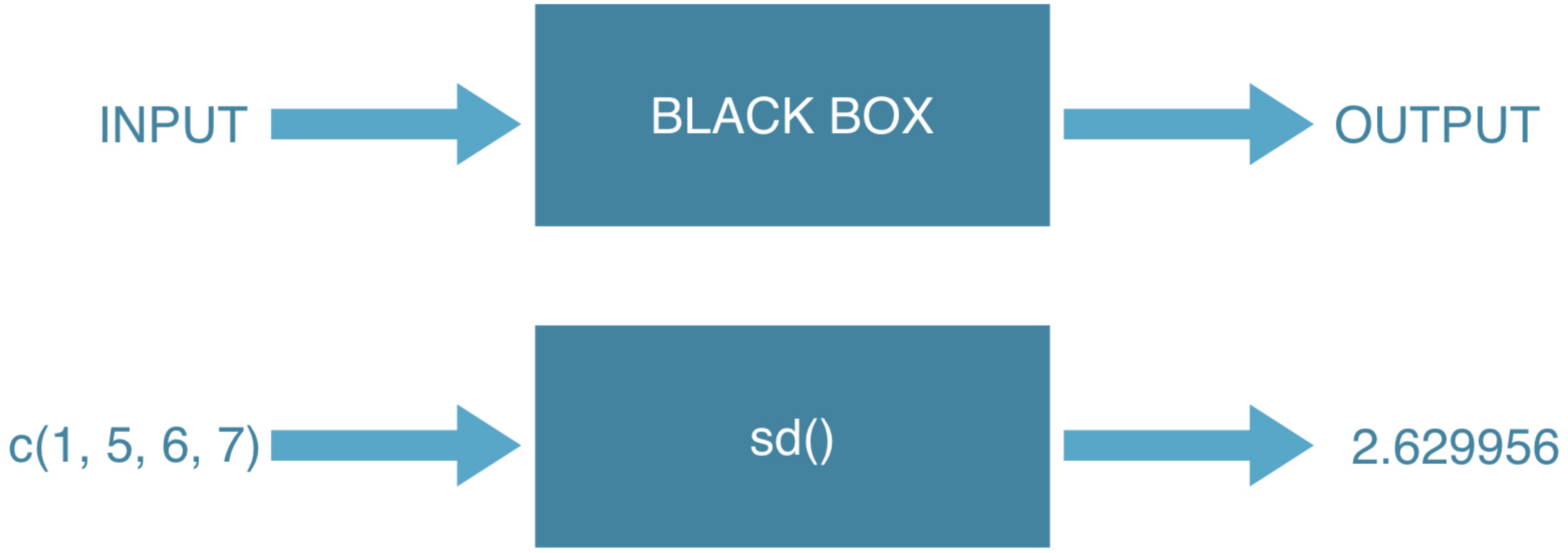
Black box principle



Black box principle



Black box principle



Call function in R

```
sd(c(1, 5, 6, 7))
```

```
2.629956
```

```
values <- c(1, 5, 6, 7)
sd(values)
```

```
2.629956
```

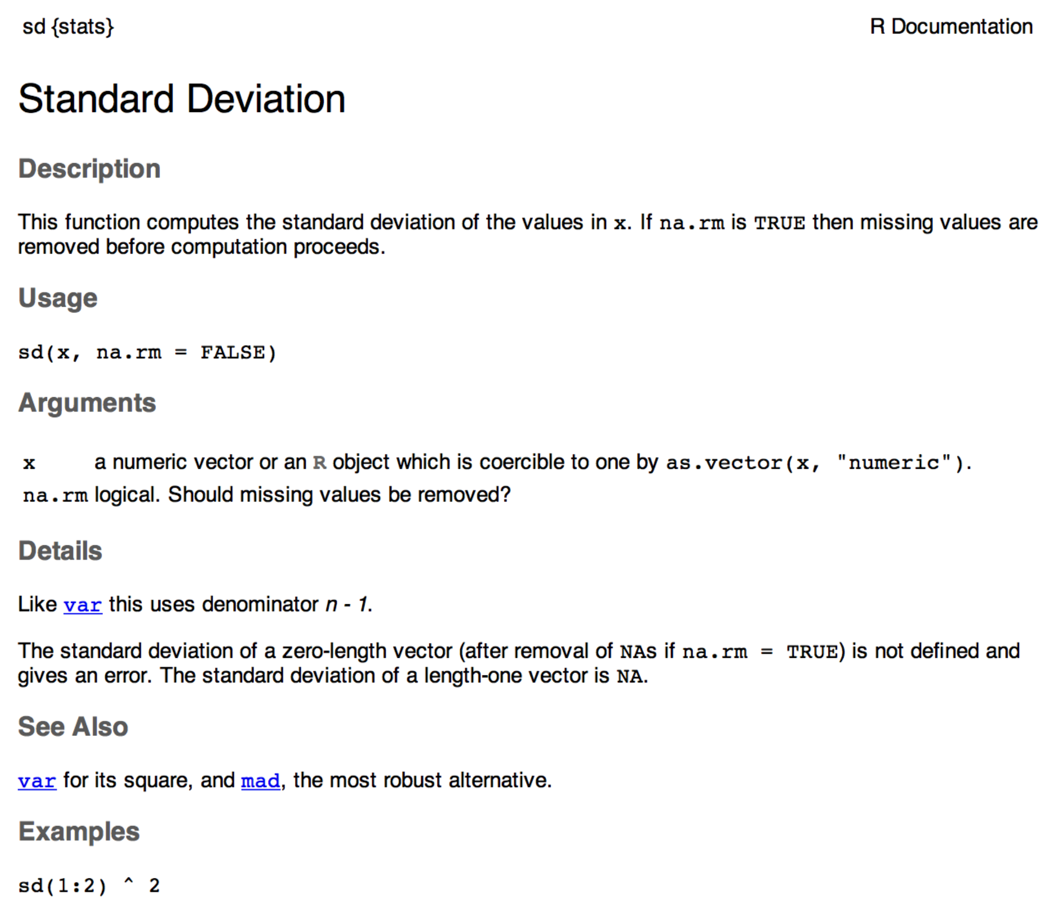
```
my_sd <- sd(values)
my_sd
```

```
2.629956
```

Function documentation

```
help(sd)
?sd
```

```
sd(x, na.rm = FALSE)
```



sd {stats} R Documentation

Standard Deviation

Description

This function computes the standard deviation of the values in `x`. If `na.rm` is `TRUE` then missing values are removed before computation proceeds.

Usage

```
sd(x, na.rm = FALSE)
```

Arguments

`x`	a numeric vector or an R object which is coercible to one by `as.vector(x, "numeric")`.
`na.rm`	logical. Should missing values be removed?

Details

Like `var` this uses denominator *n - 1*.

The standard deviation of a zero-length vector (after removal of `NA`s if `na.rm = TRUE`) is not defined and gives an error. The standard deviation of a length-one vector is `NA`.

See Also

`var` for its square, and `mad`, the most robust alternative.

Examples

```
sd(1:2) ^ 2
```

Questions

```
sd(x, na.rm = FALSE)
```

- Argument names: x, na.rm
- na.rm = FALSE
- sd(values) works?

Argument matching

```
sd(x, na.rm = FALSE)
```

By position

```
sd(values)
```

By name

```
sd(x = values)
```

na.rm argument

```r
values <- c(1, 5, 6, NA)
sd(values)
```

```
NA
```

```r
sd(x, na.rm = FALSE)
```

```r
sd(values, TRUE)
```

```
2.645751
```

```r
sd(values, na.rm = TRUE)
```

```
2.645751
```

sd(values) works?

```
values <- c(1, 5, 6, 7)
sd(values)
```

```
2.629956
```

```
sd()
```

```
Error in is.data.frame(x) :
argument "x" is missing, with no default
```

```
sd(x, na.rm = FALSE)
```

Useful trick

```r
args(sd)
```

```
function (x, na.rm = FALSE)
NULL
```

Wrap-up

- Functions work like a black box
- Argument matching: by position or by name
- Function arguments can have defaults

Let's practice!

INTERMEDIATE R

Writing Functions

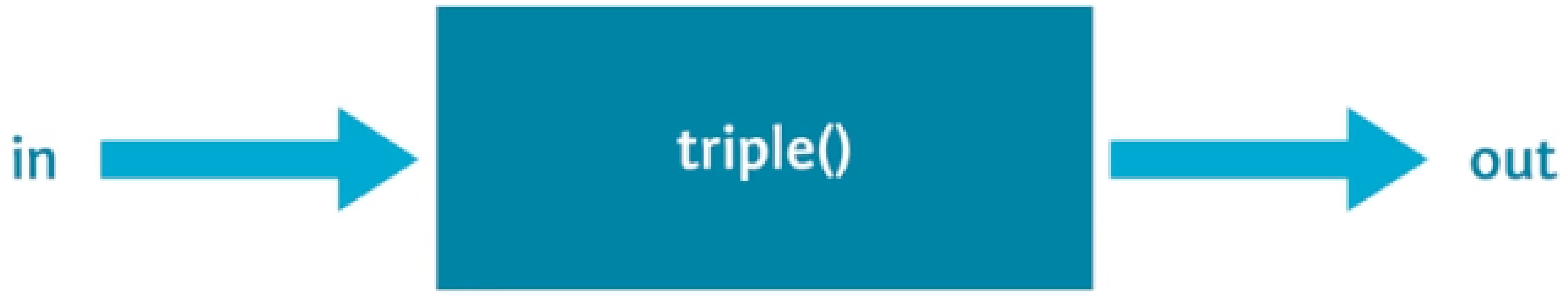
INTERMEDIATE R

Filip Schouwenaars
DataCamp Instructor

When write your own?

- Solve a particular, well-defined problem
- Black box principle
- If it works, inner workings less important

The triple() function



The triple() function

```r
my_fun <- function(arg1, arg2) {
  body
}
```

The triple() function

```r
triple <- function(arg1, arg2) {
  body
}
```

The triple() function

```r
triple <- function(x) {
  body
}
```

The triple() function

```
triple <- function(x) {
  3 * x
}
```

The triple() function

```r
triple <- function(x) {
  3 * x
}
```

```r
ls()
```

```
"triple"
```

```r
triple(6)
```

```
18
```

- Numeric 6 matched to argument x (by pos)
- Function body is executed: 3 * 6
- Last expression = return value

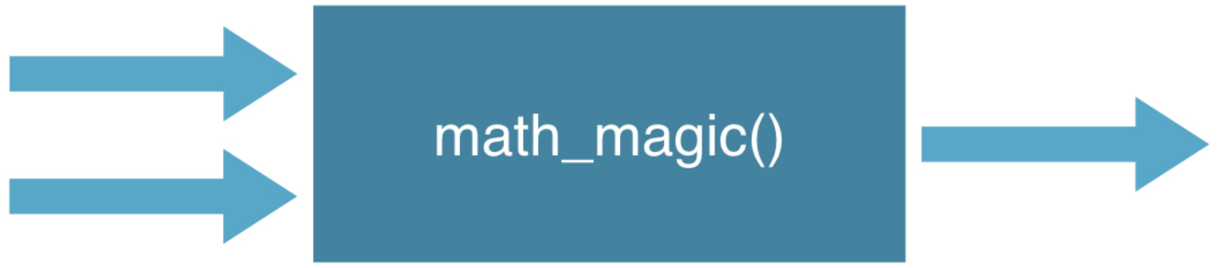
return()

```r
triple <- function(x) {
  y <- 3 * x
  return(y)
}
```

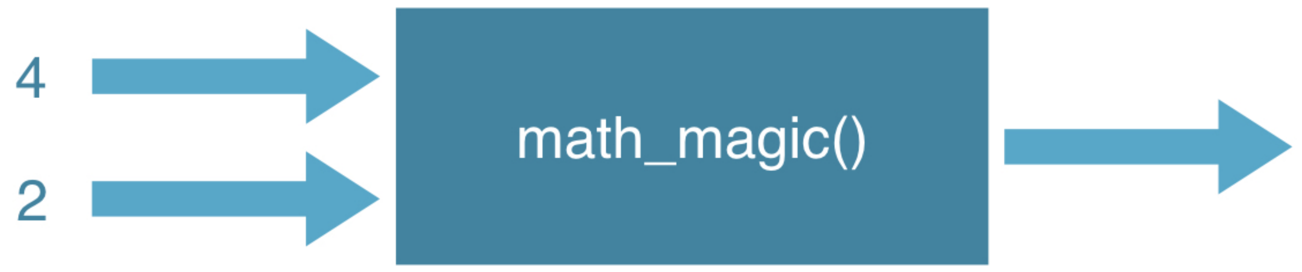
```r
triple(6)
```

```
18
```

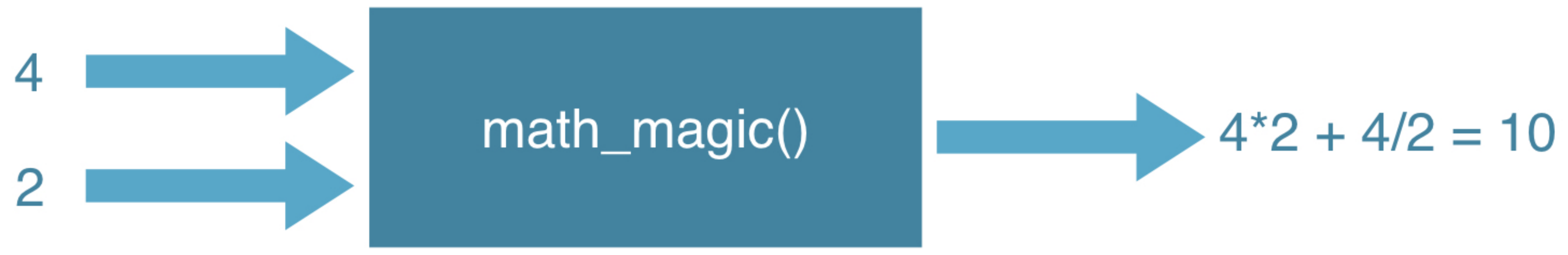
The math_magic() function



The math_magic() function



The math_magic() function



The math_magic() function

```
my_fun <- function(arg1, arg2) {
  body
}
```

The math_magic() function

```r
math_magic <- function(arg1, arg2) {
  body
}
```

The math_magic() function

```r
math_magic <- function(a, b) {
  body
}
```

The math_magic() function

```r
math_magic <- function(a, b) {
  a*b + a/b
}
```

```r
math_magic(4, 2)
```

```
10
```

```r
math_magic(4)
```

```
Error in math_magic(4) : argument "b" is missing, with no default
```

Optional argument

```
math_magic <- function(a, b = 1) {
  a*b + a/b
}
```

```
math_magic(4)
```

```
8
```

```
math_magic(4, 0)
```

```
Inf
```

Use return()

```r
math_magic <- function(a, b = 1) {
  if(b == 0){
    return(0)
  }
  a*b + a/b
}
```

```r
math_magic(4, 0)
```

```
0
```

Let's practice!

INTERMEDIATE R

R Packages

INTERMEDIATE R

Filip Schouwenaars
DataCamp Instructor

R Packages

- Where do `mean()` , `list()` and `sample()` come from?
- Part of R packages
- Code, data, documentation and tests
- Easy to share
- Examples: `base` , `ggvis`

Install packages

- `base` package: automatically installed
- `ggvis` package: not installed yet

```
install.packages("ggvis")
```

- CRAN: Comprehensive R Archive Network

Load packages

- load package = attach to search list

```
search()
```

```
".GlobalEnv"  ...  "Autoloads"   "package:base"
```

- 7 packages are attached by default
- ggvis not attached by default

```
ggvis(mtcars, ~wt, ~hp)
```

```
Error: could not find function "ggvis"
```

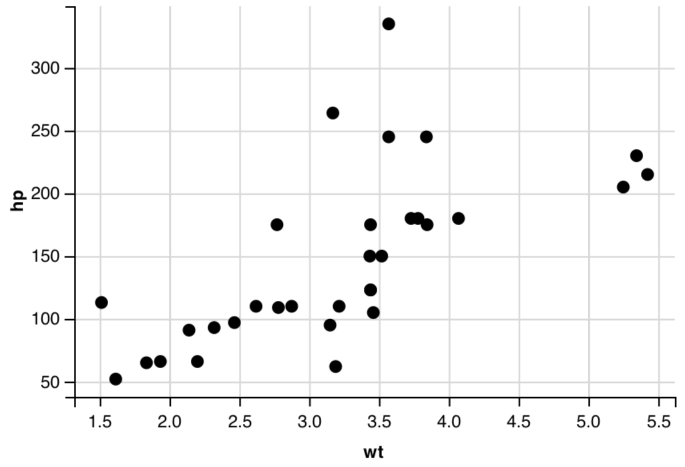
Load packages: library()

```
library("ggvis")
```

```
search()
```

```
".GlobalEnv"  "package:ggvis" ... "package:base"
```

```
ggvis(mtcars, ~wt, ~hp)
```



Load packages: require()

```r
library("data.table")
```

```
Error in library("data.table") :
there is no package called ‘data.table’
```

```r
require("data.table")
```

```
Loading required package: data.table
Warning message: ...
```

Load packages: require()

```r
result <- require("data.table")
```

```
Loading required package: data.table
Warning message: ...
```

```r
result
```

```
FALSE
```

Wrap-up

- Install packages: `install.packages()`
- Load packages: `library()` , `require()`
- Load package = attach package to search list
- Google for cool R packages!

Let's practice!

INTERMEDIATE R