Calling Functions - Scoping - Practice

In-class practice

Marcus Birkenkrahe (pledged)

January 30, 2025

README

- This file contains exercises for the lecture "Calling Functions - Scoping" in the Advanced Introduction to Data Science course (DSC205).
- Most of this material can be found in Davies, Book of R, Chapter 9. Solutions can be found in GitHub (PDF).

DONE Identify and pledge yourself

1. In Emacs, replace the placeholder `[yourname]` at the top of this file by your own name and write `(pledged)` next to it
2. Go with the cursor on the headline and hange the `TODO` label to `DONE` by entering `S-<right>` ("Shift + right-arrow").

Display package content

1. Find the built-in and automatically loaded `methods` package, and display its name (through subsetting).

   Tip: `search()` is a `character` vector of `character` vectors. Since it's a vector, you can subset it.

   Output:

   ```
   : [1] "package:methods"
   ```

   Solution:

```
search() # all loaded packages
search()[search()=="package:methods"] # methods package only
```

```
 [1] ".GlobalEnv"        "ESSR"              "package:stats"     "package:graphics
 [5] "package:grDevices" "package:utils"     "package:datasets"  "package:methods"
 [9] "Autoloads"         "package:base"
[1] "package:methods"
```

2. Display only the first 20 items contained in the built-in and automatically loaded `methods` package.

   Tip: `ls` returns a vector of `character` strings of the names of the objects in the specified environment.

```
ls("package:methods") |> head(20)
```

```
 [1] "addNextMethod"         "allNames"              "Arith"
 [4] "as"                    "as<-"                  "asMethodDefinition"
 [7] "assignClassDef"        "assignMethodsMetaData" "balanceMethodsList"
[10] "body<-"                "cacheGenericsMetaData" "cacheMetaData"
[13] "cacheMethod"           "callGeneric"           "callNextMethod"
[16] "canCoerce"             "cbind2"                "checkAtAssignment"
[19] "checkSlotAssignment"   "classesToAM"
```

1. How many items are there in total?

```
ls("package:methods") -> package_methods
length(package_methods)
```

```
[1] 203
```

Functions and environments

1. Which R `environment` owns the `read.table` function?

```
environment(read.table)
```

```
<environment: namespace:utils>
```

2. Which `environment` owns the `data` function?

   ```
   environment(data)
   ```

   ```
   <environment: namespace:utils>
   ```

3. Which environment owns the `matrix` function?

   ```
   environment(matrix)
   ```

   ```
   <environment: namespace:base>
   ```

4. Which environment owns the `jpeg` function?

   ```
   environment(png)
   ```

   ```
   <environment: namespace:grDevices>
   ```

5. What does `png` do? To find this out, run the `help` function on the command on the R console, and then use the function.

   Tip: `hist(Nile)` creates a simple histogram.

   Solution: `png` saves a plot in a graphics device (e.g. a file)

   - Using the last example in the `help` page:

     ```
     png(file="../img/example.png")
     example(rect)
     dev.off()
     system("file ../img/example.png")
     ```

     ```
     require(grDevices)
     ## set up the plot region:
     op <- par(bg = "thistle")
     plot(c(100, 250), c(300, 450), type = "n", xlab = "", ylab = "",
     rect+      main = "2 x 11 rectangles; ’rect(100+i,300+i,  150+i,380+i)’")
     i <- 4*(0:10)
     ## draw rectangles with bottom left (100, 300)+i
     ## and top right (150, 380)+i
     ```

```
rect(100+i, 300+i, 150+i, 380+i, col = rainbow(11, start = 0.7, end = 0.1))
rect(240-i, 320+i, 250-i, 410+i, col = heat.colors(11), lwd = i/5)
## Background alternating  ( transparent / "bg" ) :
j <- 10*(0:5)
rect(125+j, 360+j,   141+j, 405+j/2, col = c(NA,0),
rect+      border = "gold", lwd = 2)
rect(125+j, 296+j/2, 141+j, 331+j/5, col = c(NA,"midnightblue"))
mtext("+  2 x 6 rect(*, col = c(NA,0)) and  col = c(NA,\"m..blue\")")
## an example showing colouring and shading
plot(c(100, 200), c(300, 450), type= "n", xlab = "", ylab = "")
rect(100, 300, 125, 350) # transparent
rect(100, 400, 125, 450, col = "green", border = "blue") # coloured
rect(115, 375, 150, 425, col = par("bg"), border = "transparent")
rect(150, 300, 175, 350, density = 10, border = "red")
rect(150, 400, 175, 450, density = 30, col = "blue",
rect+      angle = -30, border = "transparent")
legend(180, 450, legend = 1:4, fill = c(NA, "green", par("fg"), "blue"),
rect+        density = c(NA, NA, 10, 30), angle = c(NA, NA, 30, -30))
par(op)
null device
  1
../img/example.png: PNG image data, 480 x 480, 8-bit/color RGB, non-interlace
```

- Using `hist(Nile)` as a simple plot to be saved.

  ```
  png(file="../img/hist.png")
  hist(Nile)
  dev.off()
  system("file ../img/hist.png")
  ```

  ```
  null device
            1
  ../img/hist.png: PNG image data, 480 x 480, 8-bit colormap, non-interlaced
  ```

6. Show that `base::matrix` is called **after** `utils::read.table` by comparing the indices in the `character` vector `search()`.

   Tip: to extract indices, you can use the `which` function in connection with logical operators.

   ```
   which(search()=="package:utils") < which(search()=="package:base")
   ```

   ```
   [1] TRUE
   ```

Functions and packages

Use `ls` and a test for `character` string equality to confirm that the function `smoothScatter` is part of the `graphics` package.

Tip: given a set of logical vectors, the function `any` tests if at least one of the values is true. E.g.

```
any(c("Jim","Jane","Joe") == "Jane")  # this is TRUE
any(c("Jim","Jane","Joe") == "Janet")  # this is FALSE
```

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

```
any(ls("package:graphics")=="smoothScatter")
```

```
[1] TRUE
```

DONE Upload completed Org-file to Canvas

Well done! You've reached the end of this in-class practice file.

If you've completed all steps, you upload the Org-mode file to Canvas (see `Assignments > Classroom Assignments`).