R4DS

Hennie Bezuidenhout

03 October 2016

# Programming in R Studio - Pipes

## The point of the **Pipe** is to help us read our code in an easy to understand way!!

### magrittr package - but all packages in tidyverse automatiically make

When are we better off not using the *Pipe* ?

1. When **Pipes** are longer than ten steps (Rather create intermediate objects with meaningful names)
2. We have multiple *inputs* or *outputs* (the same solution as above applies)
3. When we are starting to think about a *directed graph* with a complex dependency structure

Remember that the%T>%can assist us with more complex *Pipes* -Effectively, the **T-Pipe** returns the left-hand side instead of the right-hand side incase we want to plot or print

# Introducing Functions

## Allows us to automate common tasks in a more powerful and general way than copy-and-pasting



We do not copy and paste

What are the advantages of using *functions* ?

1. You can give a function an evocative name that makes your code easier to understand.
2. As requirements change, you only need to update code in one place, instead of many.
3. You eliminate the chance of making incidental mistakes when you copy and paste (i.e. updating a variable name in one place, but not in another)

##### We should seriously consider writing a function whenever we have copied and pasted a block of code more than **TWICE**

There are 3 steps to creating a new *function* ?

1. You need to pick a name for the function
2. You list the inputs, or arguments, to the function inside function
3. You place the code you have developed in **body** of the function

rescale01 <- function(x) { rng <- range(x, na.rm = TRUE) (x - rng[1]) / (rng[2] - rng[1]) }

Most important attribute of functions: \*\* WE DO NOT WANT TO REPEAT OURSELVES\*\*

## Conditional Execuion

An if statement allows us to conditionally execute code

if (condition) {  
# code executed when the condition is \*\*TRUE\*\*  
}  
else {  
# code executed when the condition is \*\*FALSE\*\*  
}