How to find help

## Reading Help files

R, and every package, provides help files for functions. To search for help on any function use a **?** before the “function\_name”:

?function\_name  
help(function\_name)

This will load up a help page in RStudio (or as plain text in R by itself).

Each help page is broken down into sections:

* Description: An extended description of what the function does.
* Usage: The arguments of the function and their default values.
* Arguments: An explanation of the data each argument is expecting.
* Details: Any important details to be aware of.
* Value: The data the function returns.
* See Also: Any related functions you might find useful.
* Examples: Some examples for how to use the function.

Different functions might have different sections, but these are the main ones you should be aware of.

## Tip: Running Examples

From within the function help page, you can highlight code in the Examples and hit Ctrl+Return to run it in RStudio console. This is gives you a quick way to get a feel for how a function works.

## Tip: Reading help files

One of the most daunting aspects of R is the large number of functions available. It would be prohibitive, if not impossible to remember the correct usage for every function you use. Luckily, the help files mean you don’t have to!

## Special Operators

To seek help on special operators, use quotes:

?"<-"

## Getting help on packages

Many packages come with “vignettes”: tutorials and extended example documentation. Without any arguments, vignette() will list all vignettes for all installed packages; vignette(package="package-name") will list all available vignettes for package-name, and vignette("vignette-name") will open the specified vignette.

If a package doesn’t have any vignettes, you can usually find help by typing help("package-name").

## When you kind of remember the function

If you’re not sure what package a function is in, or how it’s specifically spelled you can do a fuzzy search:

??function\_name

## When you have no idea where to begin

If you don’t know what function or package you need to use [CRAN Task Views](http://cran.at.r-project.org/web/views) is a specially maintained list of packages grouped into fields. This can be a good starting point.

## Challenge 1

Look at the help for the sum function. What are two ways you can pass numbers into the function so the are added together:

## Challenge 2

Look at the help for the paste function. You’ll need to use this later. What is the difference between the sep and collapse arguments?

The difference between sep and collapse is a little tricky. The paste function accepts any number of arguments, each of which can be a vector of any length. The sep argument specifies the string used between concatenated terms — by default, a space. The result is a vector as long as the longest argument supplied to paste. In contrast, collapse specifies that after concatenation the elements are *collapsed* together using the given separator, the result being a single string. e.g.

paste(c("a","b"), "c")

## [1] "a c" "b c"

paste(c("a","b"), "c", sep = ",")

## [1] "a,c" "b,c"

paste(c("a","b"), "c", collapse = "|")

## [1] "a c|b c"

paste(c("a","b"), "c", sep = ",", collapse = "|")

## [1] "a,c|b,c"

(For more information, scroll to the bottom of the ?paste help page and look at the examples, or try example('paste').)

## Solution to Challenge 1

Look at the help for the sum function. What are two ways you can pass numbers into the function so the are added together:

?sum  
sum(1, 2, 3, 4, 5)  
x <- c(1, 2, 3)  
sum(x)

## Solution to Challenge 2

To look at the help for the paste() function, use:

help("paste")  
?paste