# Using captioner

*January 2015*

## Using captioner

`captioner` is a lightweight R package that allows you to store figure and table captions and print them later. It also automatically numbers the captions so that you don't have to renumber each time you rearrange them. This is particularly useful for R markdown documents, which do not currently automatically number captions.

### Installation

`captioner` can be installed via Github. First, install `devtools` if you haven't already:

```
install.packages("devtools")
```

Now install `captioner`:

```
devtools::install_github("adletaw/captioner/captioner")
```

### Usage

Currently, the only function within `captioner` is `captioner()`. A call to this function will return a new function that you can then use to create and store your captions. This allows you to create a new function for each set of figures, tables, or any other things that you want to number separately.

#### Creating your first set of captions

When you create your function, you can also designate a prefix that will go before the figure number (e.g. "Table 1", "Figure 3"). The default prefix is "Figure", but you can use anything you want.

```
library(captioner)

table_nums <- captioner(prefix = "Table")
```

The function, `table_nums` takes 3 arguments: a key word to describe your table, a caption, and a logical indicating whether or not you are generating a citation. A citation generates only the prefix and table number, whereas the normal behavior is to also display the caption. The default behavior is to generate the full caption.

Now you can create your first caption:

```
table_nums(name = "iris", caption = "Edgar Anderson's iris data.  All measurements are in centimetres.")
```

```
## [1] "Table  1: Edgar Anderson's iris data.  All measurements are in centimetres."
```

The caption is automatically stored, so you can always reference it using the supplied keyword:

```
table_nums("iris")
```

```
## [1] "Table  1: Edgar Anderson's iris data.  All measurements are in centimetres."
```

If you want to use the caption in a formal R markdown caption, you can also store it in its own object and use the `fig.cap` command within your R markdown code chunk

```
iris_cap <- table_nums("iris")
```

And your chunk header might look like:

```
{r iris_table, fig.cap = iris_cap}
```

R markdown is pretty finicky about displaying captions, though, so it is probably easiest to use an in-line R code chunk to display the caption:

```
`r table_nums("iris")`
```

Which looks like:

Table 1: Edgar Anderson's iris data. All measurements are in centimetres.

And is as good as what the `fig.cap` command will do for you.

### Creating your second set of captions

Now suppose you not only have a set of tables, but also a set of figures. You don't want the numbering of the two to get mixed up, and you also don't want all of your figures to have the "Table" prefix. All you need to do is create a new `captioner()` function:

```
fig_nums <- captioner()
```

Here, you don't need to designate the `prefix` because the default is already set to "Figure". We can make a bunch of figure captions without worrying about messing up the table numbering.

```
fig_nums("apple", "Relationship between the number of apple seeds and the size of the apple.")
```

```
## [1] "Figure  1: Relationship between the number of apple seeds and the size of the apple."
```

```
fig_nums("milk_fat", "Average percentage of fat in milk from various animals.")
```

```
## [1] "Figure  2: Average percentage of fat in milk from various animals."
```

```
fig_nums("tree_height", "Distribution of heights in common restinga tree species.")
```

```
## [1] "Figure  3: Distribution of heights in common restinga tree species."
```

And now some more tables:

```
table_nums("world_pop", "World populations ordered from greatest to least.")
```

```
## [1] "Table  2: World populations ordered from greatest to least."
```

```
table_nums("dolphins", "List of dolphin species and their brain sizes.")
```

```
## [1] "Table  3: List of dolphin species and their brain sizes."
```

### Citing

Other than displaying full captions, you probably will also want to cite your figures and tables, without the caption, in your text. You can do this by setting the `cite` argument to `TRUE`.

```
fig_nums("milk_fat", cite = TRUE)
```

```
## [1] "Figure  2"
```

This works particularly well using an inline code chunk:

```
"There was no significant trend in milk fat between gramnivores" (`r fig_nums("milk_fat", cite = TRUE)`)
```

Which looks like:

There was no significant trend in milk fat between gramnivores (Figure 2).

It is possible that you will want to cite the figure before actually displaying the figure. This is not a problem, you can create the citation with or without a caption, and `captioner` will automatically figure out the numbering. The numbering is designated the first time a figure with that keyword is encountered, whether it's a citation or a full caption.

```
fig_nums("snuff", cite = TRUE)
```

```
## [1] "Figure  4"
```

```
fig_nums("bounce", "Bounce height (cm) of balls made of various types of rubber.")
```

```
## [1] "Figure  5: Bounce height (cm) of balls made of various types of rubber."
```

```
fig_nums("snuff", "Average size of snuff boxes used from 1808 to 1908.")
```

```
## [1] "Figure  4: Average size of snuff boxes used from 1808 to 1908."
```

```
fig_nums("bounce", cite = TRUE)
```

```
## [1] "Figure  5"
```

If you find the code for citing particularly cumbersome, you can shorten it by using `partial` from the package `pryr`. This allows you to preemptively fill in some of the parameters of your `captioner()` function.

For example: `citef <- partial(fig_nums, cite = T)`

Then: `citef("bounce")`

### Implementing hierarchical numbering

If you want figure numbers with more depth, say if you have a figure with subfigures, or have figure numbers attached to sections, you can implement hierarchical numbering. This will give you, for example, something like "Figure 1.1.1".

```
figs_2 <- captioner(levels = 3, type = c("n", "n", "n"), infix = "-")
figs_2("A", "The letter A in several typefaces.")
```

```
## [1] "Figure  1-1-1: The letter A in several typefaces."
```

The three new parameters tell `captioner` how many subsets of numbers you want (`levels`), whether you want a number or a character for each (`type`), and what you want separating them (`infix`). For `type`, the choices are `"n"` - a number, `"c"` - a lowercase character, or `"C"` - an uppercase character. If your vector is too short, or if you choose not to supply any `type`, `captioner` will automatically fill in the missing `type`s with numbers. The default `infix` is `.`, but you can choose anything you want.

Normally, a call to `figs_2` will only increment the final number or character in your figure number. If you want to increment an earlier value, there are two ways. First, you can set the parameter `level` inside of `figs_2`. `level` gives the number corresponding to which value you want incremented. For example, if you want to bump up from `1.1.1` to `1.2.1`:

```
figs_2("B", cite = T, level = 2)
```

```
## [1] "Figure  1-2-1"
```

The second way to do this is to use the function `bump`, introduced in `captioner` version 2.0. If you have a document with many chapters or sections, `bump` may be more convenient because it is not connected to a specific figure. Instead, you can place it after your section heading, and all subsequent figures will have the bumped up numbering.

`bump` takes as parameters your captioner function, and a numbering indicating which level you want bumped:

```
bump(figs_2, level = 1)

figs_2("C", "The letter C shown in fixed-width fonts.")
```

```
## [1] "Figure  2-1-1: The letter C shown in fixed-width fonts."
```