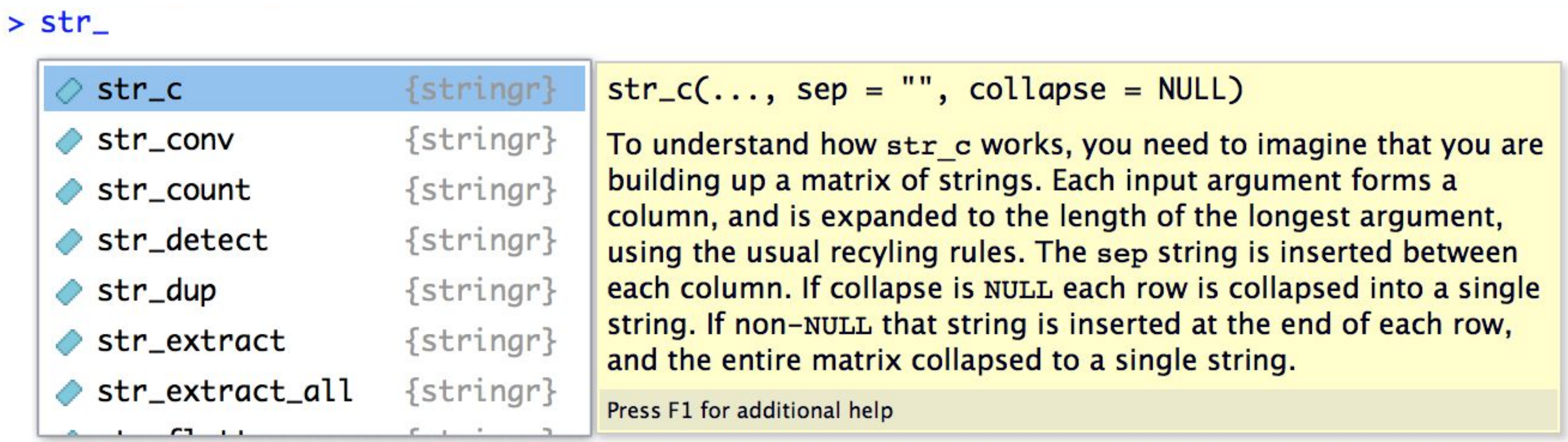
Strings

Getting and Cleaning Data

A string is a sequence of characters, letters, numbers or symbols.

stringr

www.rstudio.com



```
> str_
```

```
str_c              {stringr}
str_conv           {stringr}
str_count          {stringr}
str_detect         {stringr}
str_dup            {stringr}
str_extract        {stringr}
str_extract_all    {stringr}
```

`str_c(..., sep = "", collapse = NULL)`

To understand how `str_c` works, you need to imagine that you are building up a matrix of strings. Each input argument forms a column, and is expanded to the length of the longest argument, using the usual recyling rules. The `sep` string is inserted between each column. If collapse is `NULL` each row is collapsed into a single string. If non-`NULL` that string is inserted at the end of each row, and the entire matrix collapsed to a single string.

Press F1 for additional help

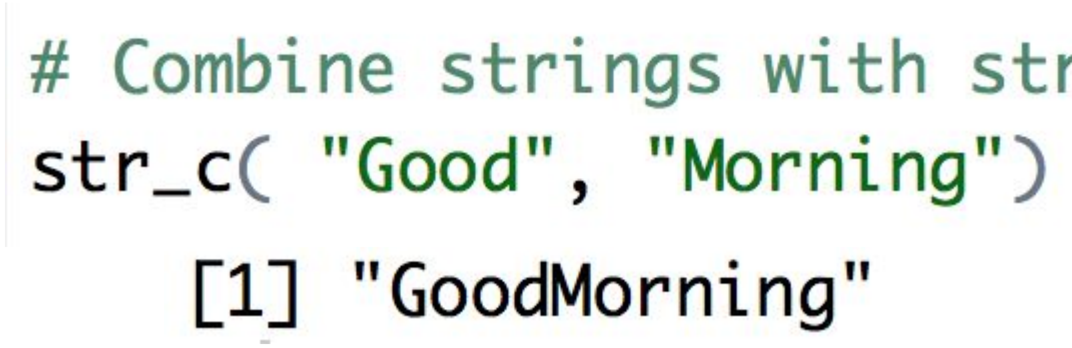
When working with strings, some of the most frequent tasks you'll need to complete are to:

- Determine the length of a string
- Combine strings together
- Subset strings
- Sort strings

```
objectA <- c( "This sentence is a string.",
              "Here's another string",
              "Here's a third string" )

# Calculate the length of strings with str_length
str_length(objectA)
```

```
    [1] 26 21 21
```



```
# Combine strings with str_c
str_c( "Good", "Morning")
    [1] "GoodMorning"
```

```r
# Use the sep argument to separate the words with a space
str_c( "Good", "Morning", sep=" ")
```

```
    [1] "Good Morning"
```

```r
# Create two strings
object <- c("Good", "Morning")

# Subset the first three characters
str_sub(object, 1, 3)
```

```
[1] "Goo" "Mor"
```

```r
# Create two strings
object <- c( "Good", "Morning")

# Subset the last three characters
str_sub(object, -3, -1)
```

```
   [1] "ood" "ing"
```

```r
names <-c("Keisha McDonald",
          "Mohammed Smith",
          "Jane Doe",
          "Mathieu Person")

# Use str_sort to sort the names alphabetically
str_sort(names)
```

```
 [1] "Jane Doe"        "Keisha McDonald" "Mathieu Person"
 [4] "Mohammed Smith"
```

```r
# Specify decreasing = TRUE to sort in reverse order
str_sort(names, decreasing = TRUE)
```

```
 [1] "Mohammed Smith"  "Mathieu Person"  "Keisha McDonald"
 [4] "Jane Doe"
```

Helpful stringr functions that can use regular expressions include:

- `str_view()` - View the first occurrence in a string that matches the regular expression
- `str_view_all()` - View all occurrences in a string that match the regular expression
- `str_count()` - Count the number of times a regular expression matches within a string
- `str_detect()` - Determine if a regular expression is found within string
- `str_subset()`- return subset of strings that match the regular expression
- `str_extract()` - return portion of each string that matches the regular expression
- `str_replace()` - replace portion of string that matches the regular expression with something else

```r
names <-c("Keisha McDonald",
          "Mohammed Smith",
          "Jane Doe",
          "Mathieu Person")

# Identify strings that start with "M"
str_view(names, "^M")
```

```
Keisha McDonald

Mohammed Smith

Jane Doe

Mathieu Person
```

```r
# Identify strings that end with "e"
str_view(names, "e$")
```

```
Keisha McDonald

Mohammed Smith

Jane Doe

Mathieu Person
```

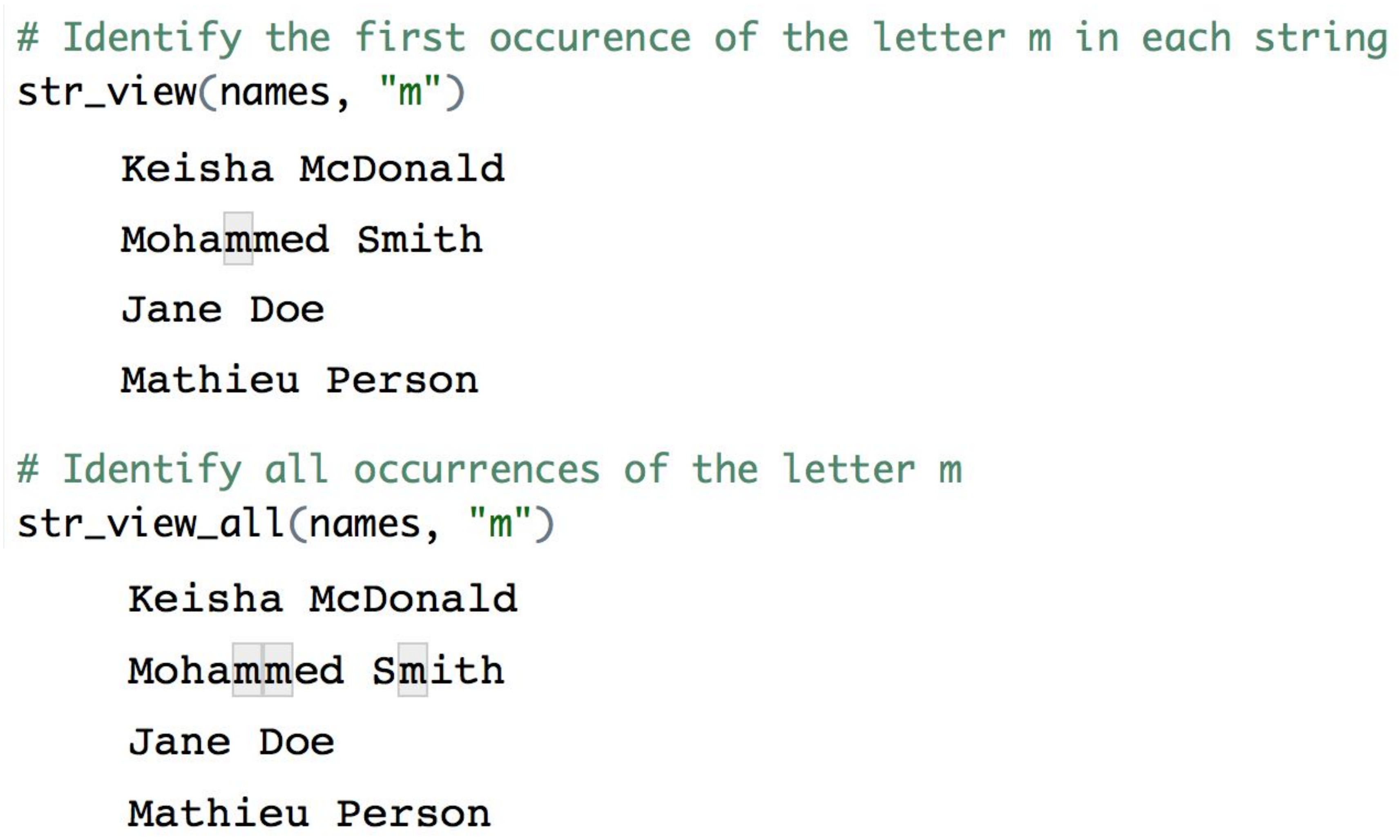
```r
# Identify strings that end with "E"
str_view(names, "E$")
```

```
Keisha McDonald

Mohammed Smith

Jane Doe

Mathieu Person
```



```
# Identify the first occurence of the letter m in each string
str_view(names, "m")
```

```
    Keisha McDonald

    Mohammed Smith

    Jane Doe

    Mathieu Person
```

```
# Identify all occurrences of the letter m
str_view_all(names, "m")
```

```
    Keisha McDonald

    Mohammed Smith

    Jane Doe

    Mathieu Person
```

```r
# Identify strings that start with "M"
# Return count of the number of times string matches pattern
str_count(names, "^M")
```

```
[1] 0 1 0 1
```

```
# Identify strings that have a lowercase "m"
# Return count of the number of times string matches pattern
str_count(names, "m")

    [1] 0 3 0 0
```

```r
# Identify strings that start with "M"
# Return TRUE if they do; FALSE otherwise
str_detect(names, "^M")
```

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

```
# Identify strings that start with "M"
# Return the whole string
str_subset(names, "^M")
   [1] "Mohammed Smith" "Mathieu Person"
```

```r
# Return "M" from strings with "M" in it
# otherwise, return NA
str_extract(names, "^M")
```

```
    [1] NA  "M" NA  "M"
```

```r
# Replace capital M with a question mark
str_replace(names, "^M", "?")
```

```
[1] "Keisha McDonald" "?ohammed Smith"
"Jane Doe"        "?athieu Person"
```

```r
# Create a vector of strings with names and their sex
names_sex <-c("Keisha McDonald, Female",
              "Mohammed Smith, male",
              "Jane Doe, female",
              "Mathieu Person, Male")

# Note the inconsistent capitalization of the sex. Let's fix that
str_replace(names_sex, "Male", "male") %>%
  str_replace("Female", "female")
```

```
[1] "Keisha McDonald, female"
[2] "Mohammed Smith, male"
[3] "Jane Doe, female"
[4] "Mathieu Person, male"
```

Summarizing: Strings

Getting and Cleaning Data