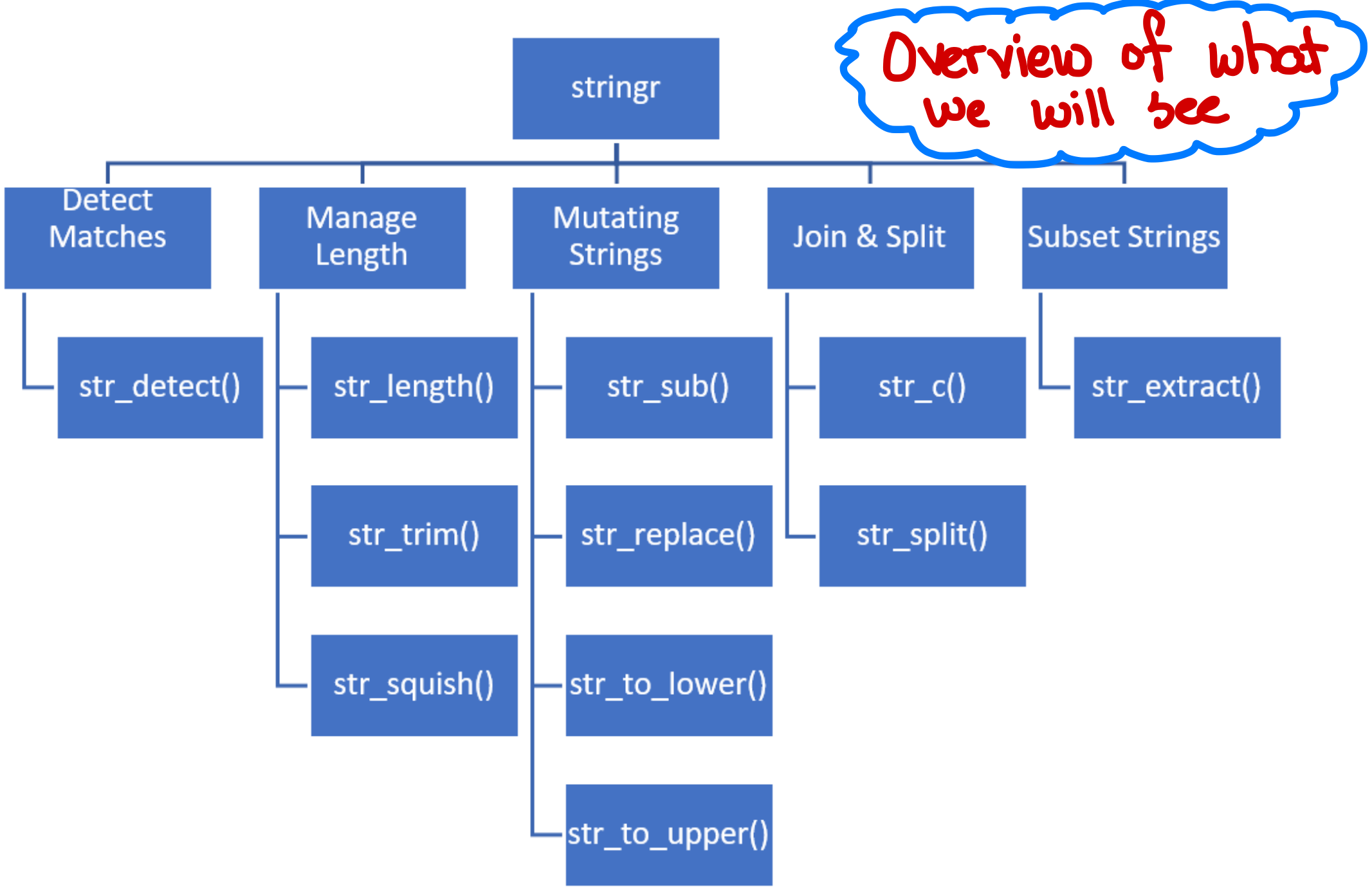
Strings


Overview of what we will see
stringr
Detect Matches
Manage Length
Mutating Strings
Join & Split
Subset Strings
str_detect()
str_length()
str_trim()
str_squish()
str_sub()
str_replace()
str_to_lower()
str_to_upper()
str_c()
str_split()
str_extract()

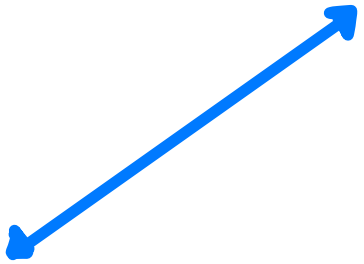
Detect Matches

- `str_detect()`: Detects Patterns - checks if a string contains a specific pattern. Output: TRUE/FALSE.

Example 1: Detect whether the word fox is in the string or not.

```
my_string <- "The quick brown fox jumps over the
lazy dog."

str_detect(my_string, "fox")
```



Output: TRUE

How can I use these functions in a dataframe?
Typically you want to use mutate.

illustration_data

Example
"the quick brown fox jumps over the lazy dog."
"hello, world!"
"fox and cat"

Example 2: Use str_detect in illustration data above to detect whether on each sentence there is the word fox.

```
example_2 <- illustration_data %>%
  mutate(fox_check = str_detect(Example, "fox"))
```

creates a new column

Example	fox_check
"the quick brown fox jumps over the lazy dog."	TRUE
"hello, world!"	FALSE
"fox and cat"	TRUE

Manage Length

- `str_length()`: Finds the length (the number of characters) in a string.

Example 3: Count how many characters are in the following string.

```
my_string <- "Hello, World!"
str_length(my_string)
```

Output: 13

- `str_trim()`: Trims white space - removes leading and trailing white space from a string.

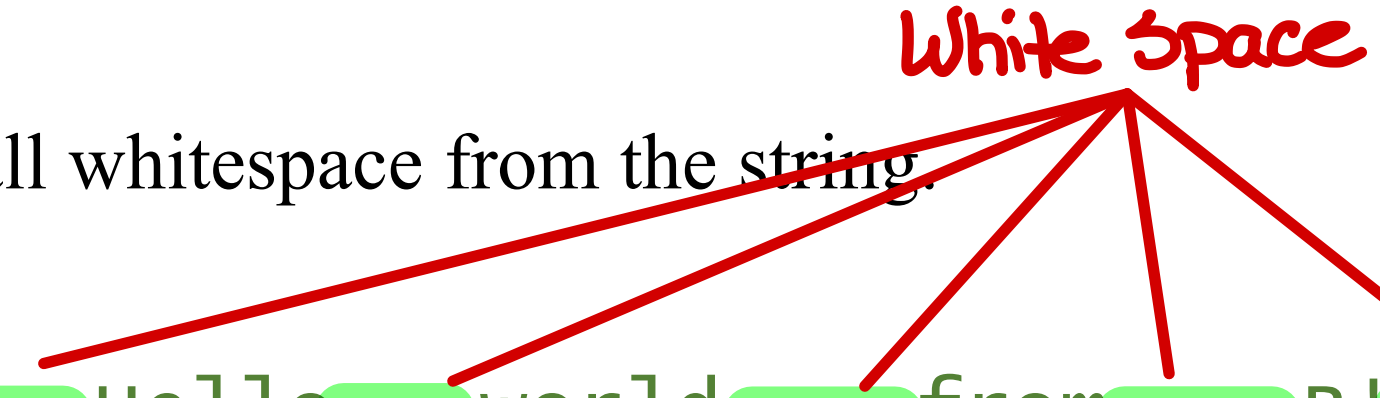
Example 4: Remove the white space from the beginning and the end of the string.


```
my_string <- "   Trim me!   "
str_trim(my_string)
```

Output: "Trim me!"

- `str_squish()`: Removes extra white space within a string, as well as the beginning, and the end.

Example 5: Remove all whitespace from the string.




```
my_string <- "   Hello   world   from   R!   "
str_squish(my_string)
```

Output: "Hello world from R!"

Mutating Strings

- `str_sub()`: extracts substrings from a string.

Example 6: Extract the string from position 8 to position 13.

```
my_string <- "Extract this part"
str_sub(my_string, start = 9, end = 13)
```

Output:

- `str_replace()`: replaces a pattern with another string.

Example 7: Substitute the word fox with the word cat.

```
my_string <- "The quick brown fox jumps over the
lazy dog."

str_replace(my_string, "fox", "cat")
```

replaces the 1st matched.

Note: str_replace_all replaces all matched patterns.

the one you want to change

the one you want to change it for.

Output: "The quick brown cat jumps over the lazy dog"

- `str_to_lower()`: Changes the case to all lower case of a string.
- `str_to_upper()`: Changes the case to all upper case of a string.

Example 8: Change the following string to all lower case and then to all upper.

```
my_string <- "Hello, World!"
```

```
str_to_lower(my_string)
```

Output: "hello, world!"

```
str_to_upper(my_string)
```

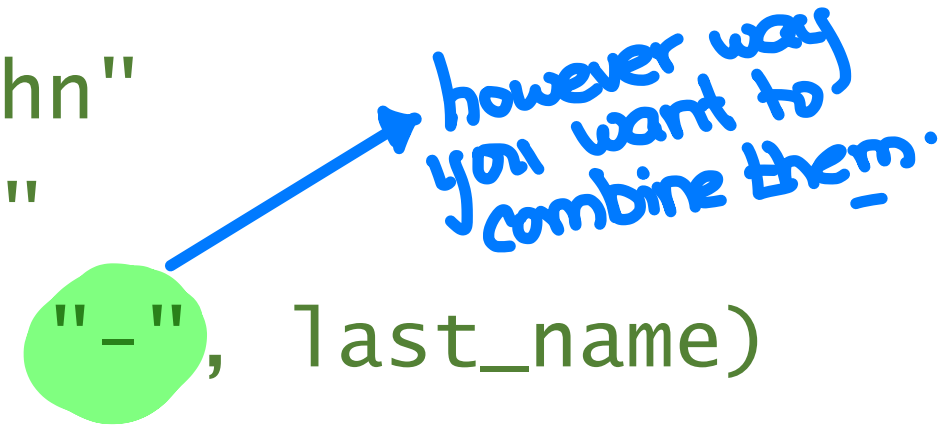
Output: "HELLO, WORLD!"

Join & Split

- `str_c()`: Combining Strings - concatenates strings together.

Example 9: combine the following two strings. Separate them by "-".

```
first_name <- "John"
last_name <- "Doe"
str_c(first_name, "-", last_name)
```



Output: "John-Doe"

- `str_split()`: splits a string into a character vector using a specified delimiter.

Example 10: Separate the following string by ",".

```
my_string <- "apple,banana,cherry"

str_split(my_string, ",")
```

Output: "apple" "banana" "cherry"

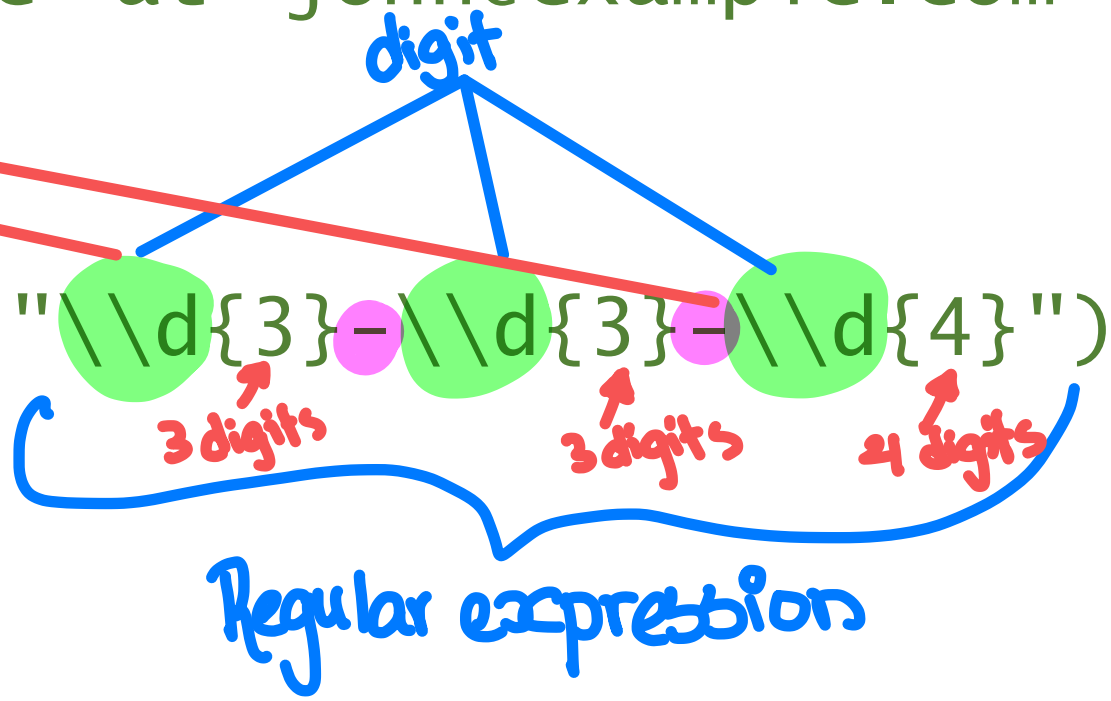
Subset Strings

- `str_extract()`: extracts the first occurrence of a pattern from a string.

Example 11: Extract the phone number from the following string.

```
my_string <- "Email me at john@example.com or
call at 555-123-4567."

str_extract(my_string, "\\d{3}-\\d{3}-\\d{4}")
```



Output: "555-123-4567"