Manipulating Strings: Brief Summary

Base R

Syntax (String = character string)	Description
`nchar(String)`	number of characters
`substr(String, start.m, end.n)`	substring from m to n
`strsplit(String, "w")`	split expression at (character) w
`sub("old", "new", String)`	Replace single instance of "old" with "new"
`gsub("old", "new", String)`	Replace instances of "old" with "new"

library(stringr)

Command	Description
`str_length(String)`	number of characters (similar to `nchar`.)
`str_sub(String, start.m, end.n)`	substring from m to n similar to `substr`.
`str_split(String, "w")`	split expression at (character) w
`str_detect(String, "pattern")`	detect presence/absence of pattern (return T/F)
`str_locate(String, "pattern")`	locate first position of pattern, return a matrix with columns start and end.
`str_locate_all(String, "pattern")`	locates all matches of pattern in string
`str_extract(String, "pattern")`	extracts text corresponding to first match
`str_extract_all(String, "pattern")`	extracts all matches and returns a list
`str_replace(String, "old", "new")`	Replace single instance of "old" with "new" (similar to `sub`).
`str_replace_all(String, "old", "new")`	Replace instances of "old" with "new" (similar to `gsub`.)

Regular Expressions

Metacharacters:

```
.   ^   \   $   ?   *   +   [   ]   (   )   {   }   |
```

If you actually want to use these symbols literally, then **in** `R`, you precede them by a double slash (or surround them with brackets).

```
str_detect(string, "\\.")  #to match a period.
str_detect(string, "[.]")  #same
```

Note that outside of `R` , a single backslash works: `\$` to match the dollar sign.

Character sets

To match one of several options, use the brackets:

```
str_detect(string, "Ch[aio]mp")
```

matches Champ, Chimp, Chomp, Champion, Chimps, ...

Anchors

Use ^ to indicate a pattern that starts a string, $ the end.

```
str_detect(String, "^Hello\\s")   #string starts with hello (followed by space)
str_detect(String,"goodbye\\.$")  #string ends with goodbye (followed by period)
```

Syntax	Description
\\d	Digit, 0,1,2 ... 9
\\D	Not Digit
\\s	Space
\\S	Not Space
\\w	Word
\\W	Not Word
\\t	Tab
\\n	New line
^	Beginning of the string
$	End of the string
\	Escape special characters, e.g. \\is "\", \+ is "+"
|	Alternation match. e.g. /(e| d)n/ matches "en" and "dn"
.	Any character, except \n or line terminator
[ab]	a or b
[^ab]	Any character except a and b
[3-5]	Range: for example, 3, 4, 5
[C-E]	Range: for example C, D, E
[d-g]	Range: for example, d, e, f or g
[A-z]	Range: Uppercase and lowercase a to z letters
s+	s at least one time
s*	s zero or more times
s?	s zero or 1 time
s{n}	s occurs n times in sequence
s{n1,n2}	s occurs between n1 and n2 times in sequence
s{n1,n2}?	non greedy match, see above example
s{n,}	s occurs n or more times
()	grouping
.*?A	match up to the first occurrence of A.

Some other useful patterns:
`str_replace(String, "patternA(patternB)patternC", "\\1")`
match above pattern and replace with patternB.