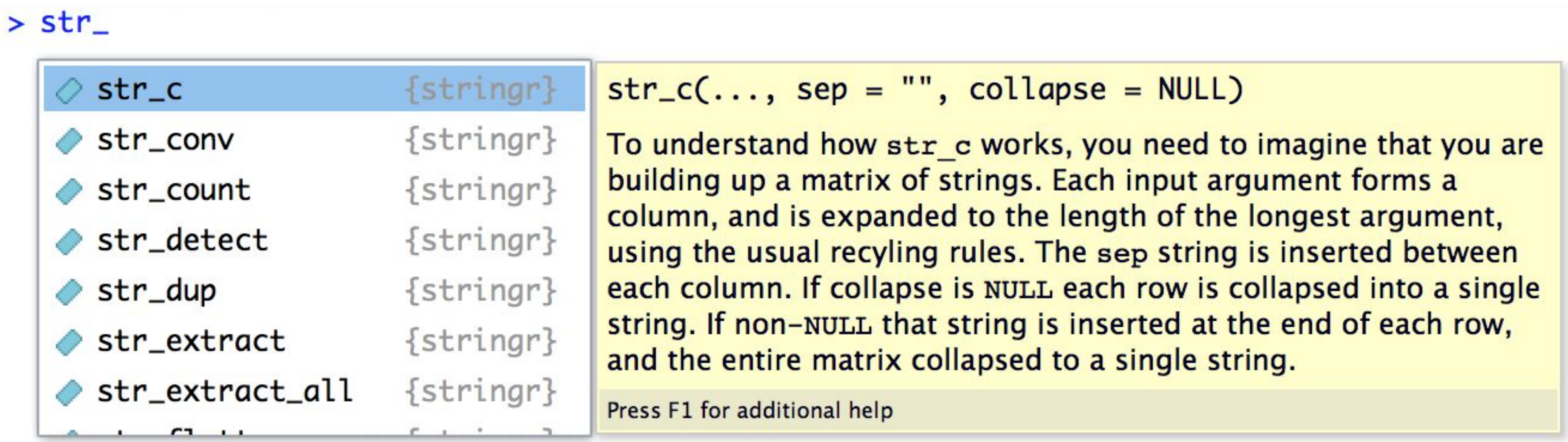
Working with: strings

Data Tidying

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

stringr

www.rstudio.com



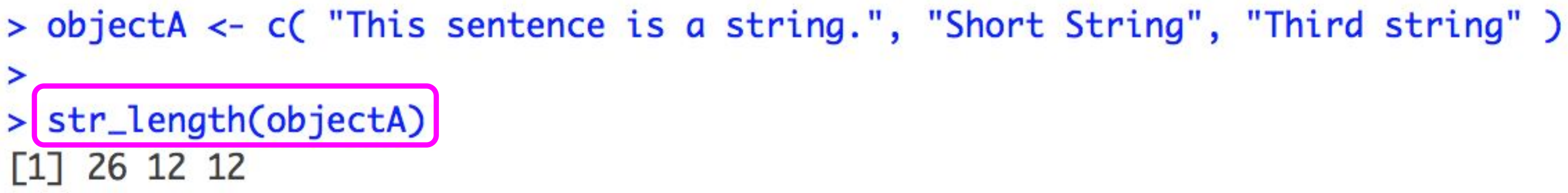
```
> str_
```

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

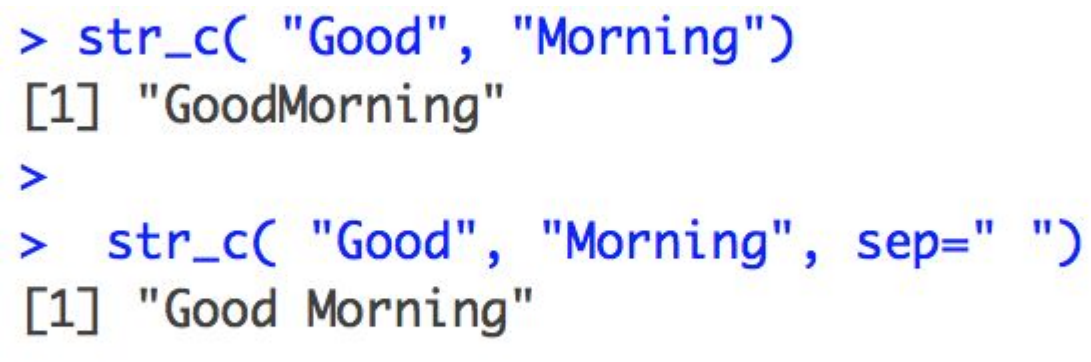
```
str_c(..., sep = "", collapse = NULL)
```

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.

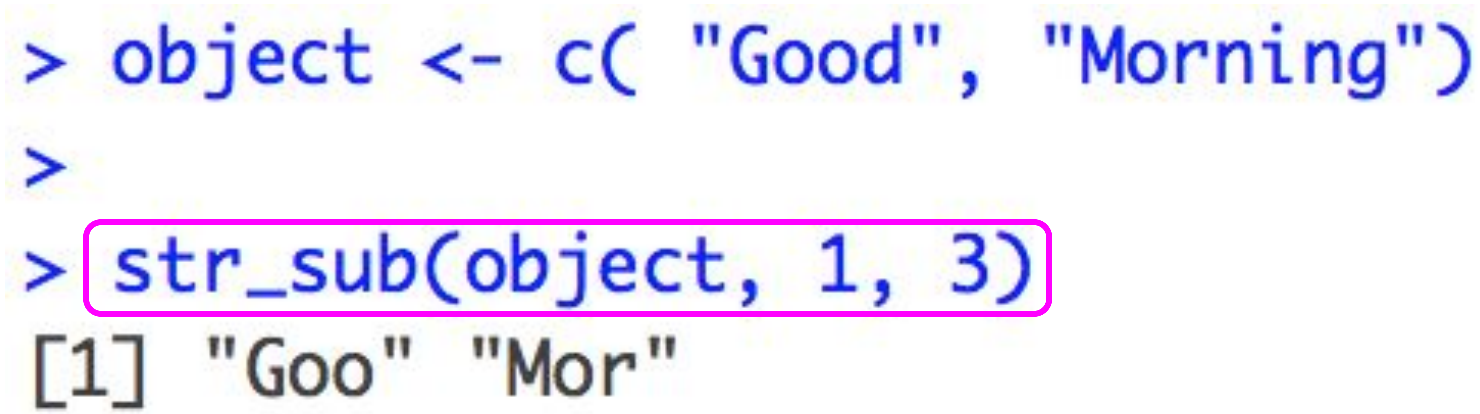
Press F1 for additional help



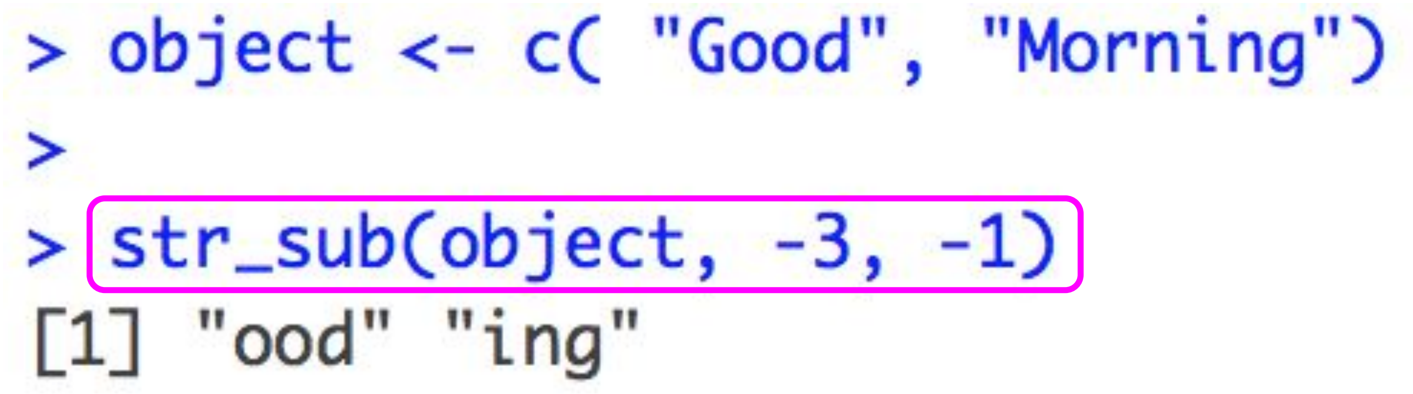
```
> objectA <- c( "This sentence is a string.", "Short String", "Third string" )
>
> str_length(objectA)
[1] 26 12 12
```



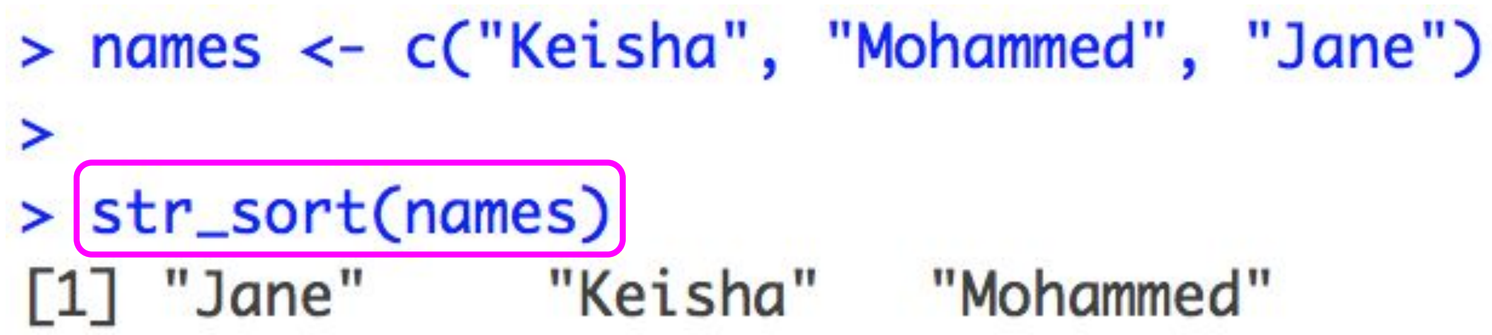
```
> str_c( "Good", "Morning")
[1] "GoodMorning"
>
>  str_c( "Good", "Morning", sep=" ")
[1] "Good Morning"
```



```
> object <- c( "Good", "Morning")
>
> str_sub(object, 1, 3)
[1] "Goo" "Mor"
```



```
> object <- c( "Good", "Morning")
>
> str_sub(object, -3, -1)
[1] "ood" "ing"
```

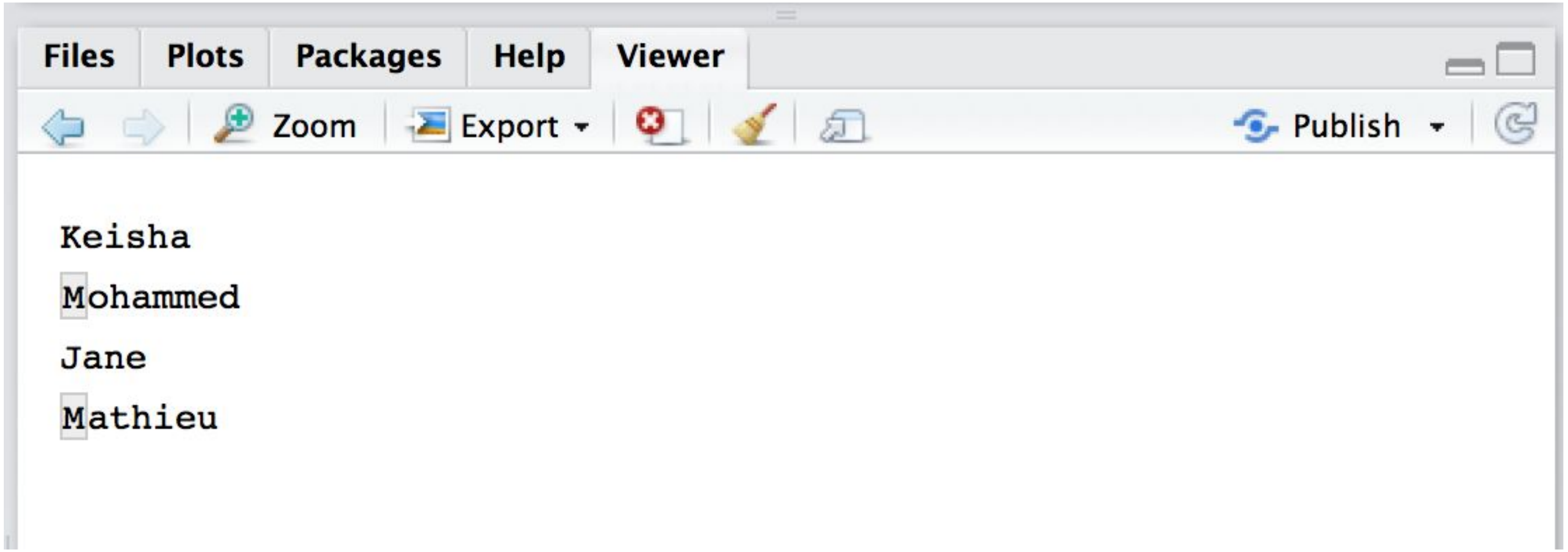


```
> names <- c("Keisha", "Mohammed", "Jane")
>
> str_sort(names)
[1] "Jane"     "Keisha"   "Mohammed"
```

Regular Expressions

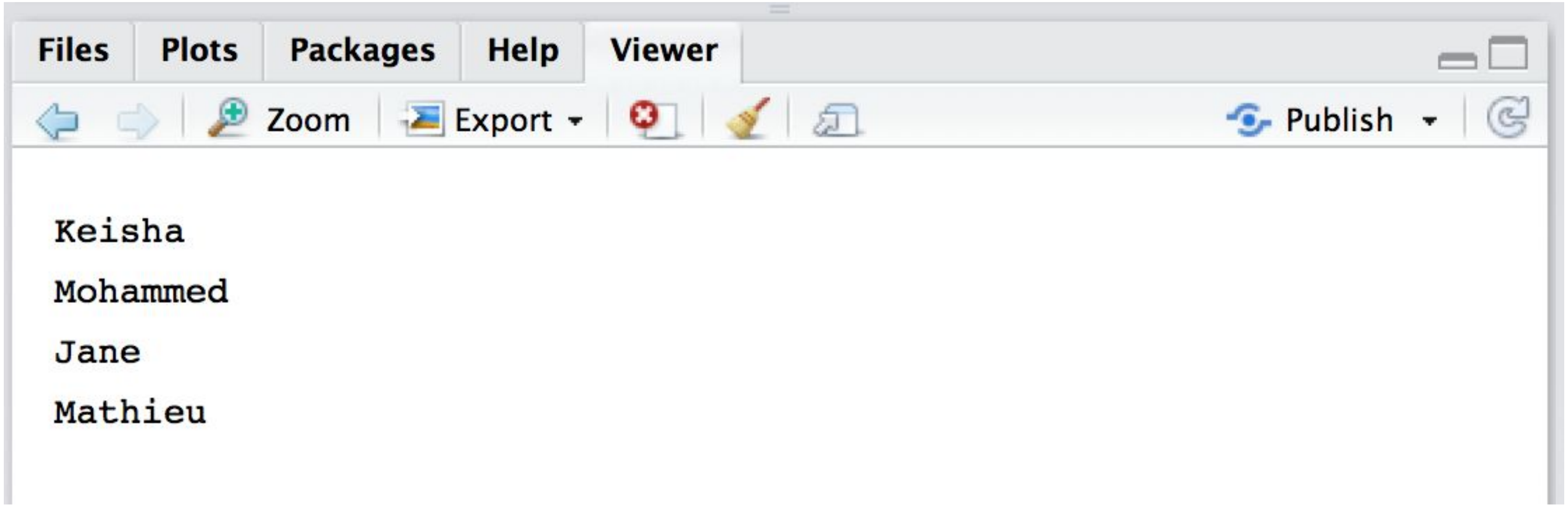
```
function(string , pattern = regexp)
```

```
names <- c("Keisha", "Mohammed", "Jane", "Mathieu")

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



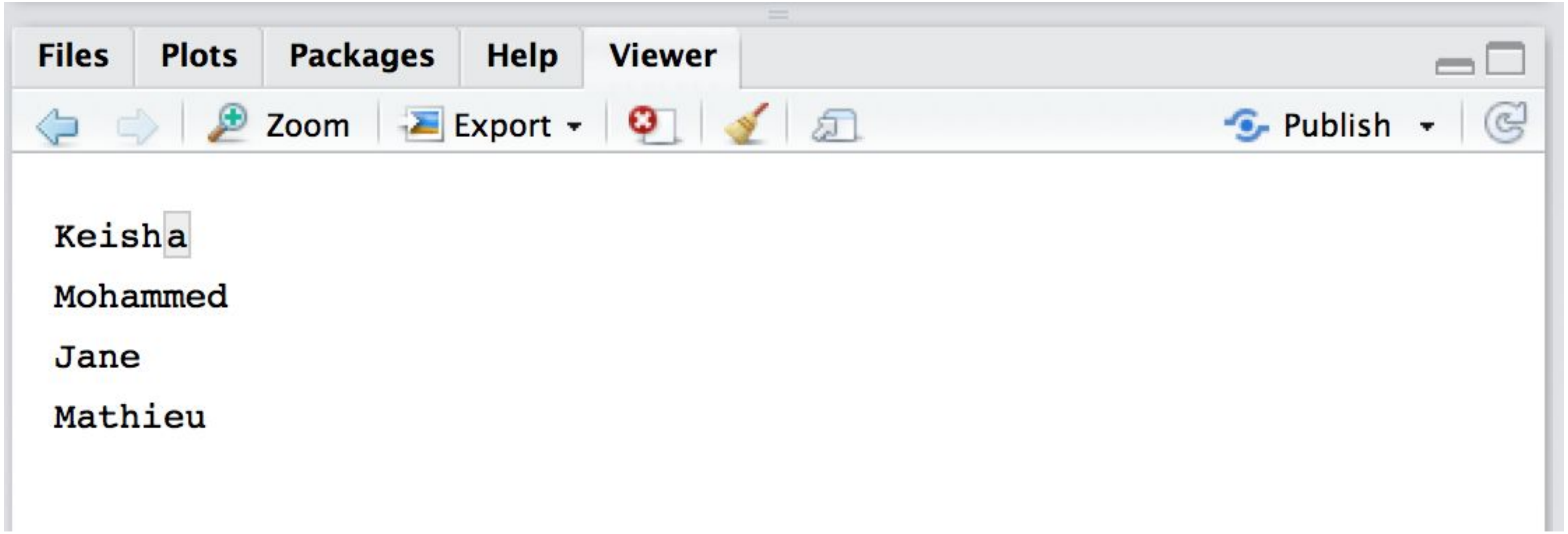
```
Keisha
Mohammed
Jane
Mathieu
```

```
names <- c("Keisha", "Mohammed", "Jane", "Mathieu")

## identify strings that end with "M"
str_view(names, "M$")
```



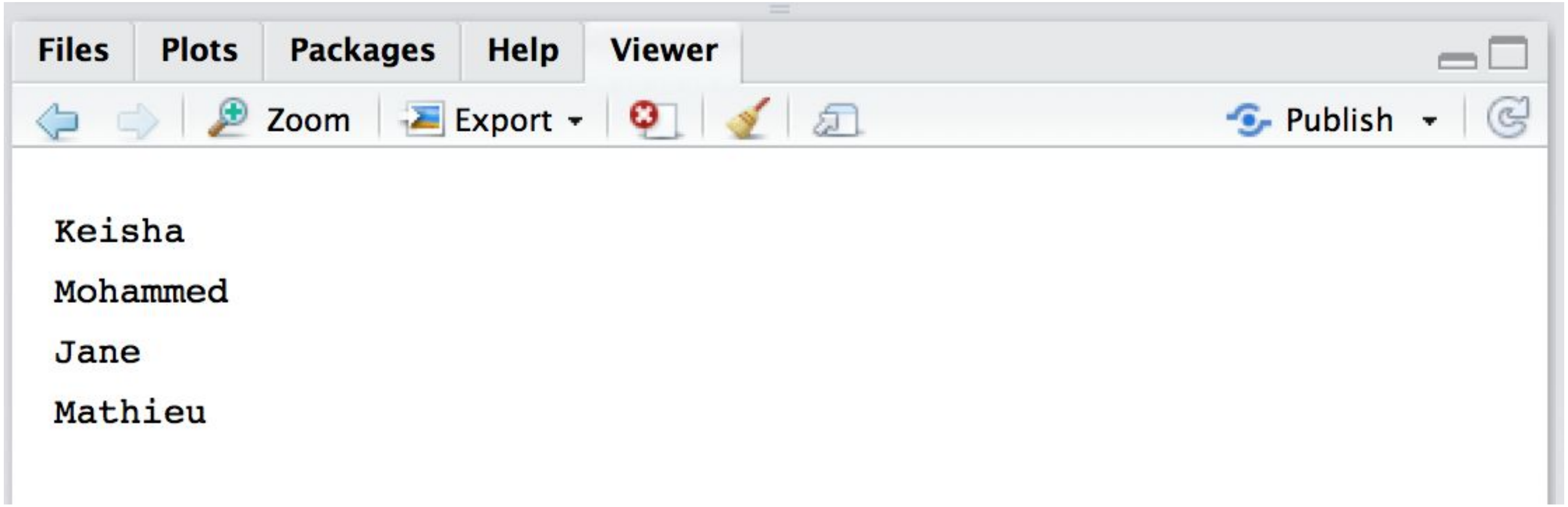
```
Keisha
Mohammed
Jane
Mathieu
```

```
names <- c("Keisha", "Mohammed", "Jane", "Mathieu")

## identify strings that end with "a"
str_view(names, "a$")
```



```
Keisha
Mohammed
Jane
Mathieu
```

```
names <- c("Keisha", "Mohammed", "Jane", "Mathieu")

## identify strings that end with "A"
str_view(names, "A$")
```



```
Keisha
Mohammed
Jane
Mathieu
```

```
names <- c("Keisha", "Mohammed", "Jane", "Mathieu")

    ## 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
```

```
names <- c("Keisha", "Mohammed", "Jane", "Mathieu")

 ## identify strings that have a lowercase "m"
 ## return count of the number of times string
 matches pattern

 > str_count(names, "m")
 [1] 0 2 0 0
```

```
names <- c("Keisha", "Mohammed", "Jane", "Mathieu")

    ## identify strings that start with "M"
    ## return TRUE if they do; FALSE
    otherwise

    > str_detect(names, "^M")
    [1] FALSE  TRUE FALSE  TRUE
```

```
names <- c("Keisha", "Mohammed", "Jane", "Mathieu")

    ## identify strings that start with "M"
    ## return whole string

    > str_subset(names, "^M")
    [1] "Mohammed" "Mathieu"
```

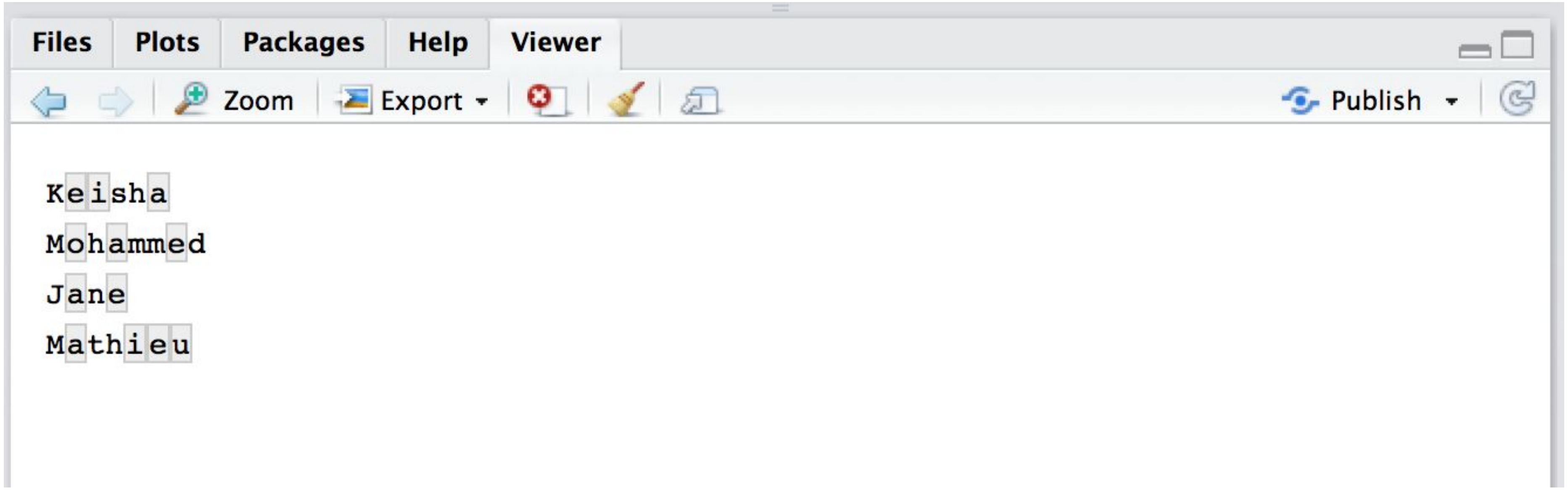
```
names <- c("Keisha", "Mohammed", "Jane", "Mathieu")

    ## return "M" from strings with "M" in it
    ## otherwise, return NA


    > str_extract(names, "^M")
    [1] NA  "M" NA  "M"
```

```
names <- c("Keisha", "Mohammed", "Jane", "Mathieu")

 ## replace capital M with a question mark

 > str_replace(names, "^M", "?")
 [1] "Keisha"   "?ohammed" "Jane"     "?athieu"
```

Common regular expressions

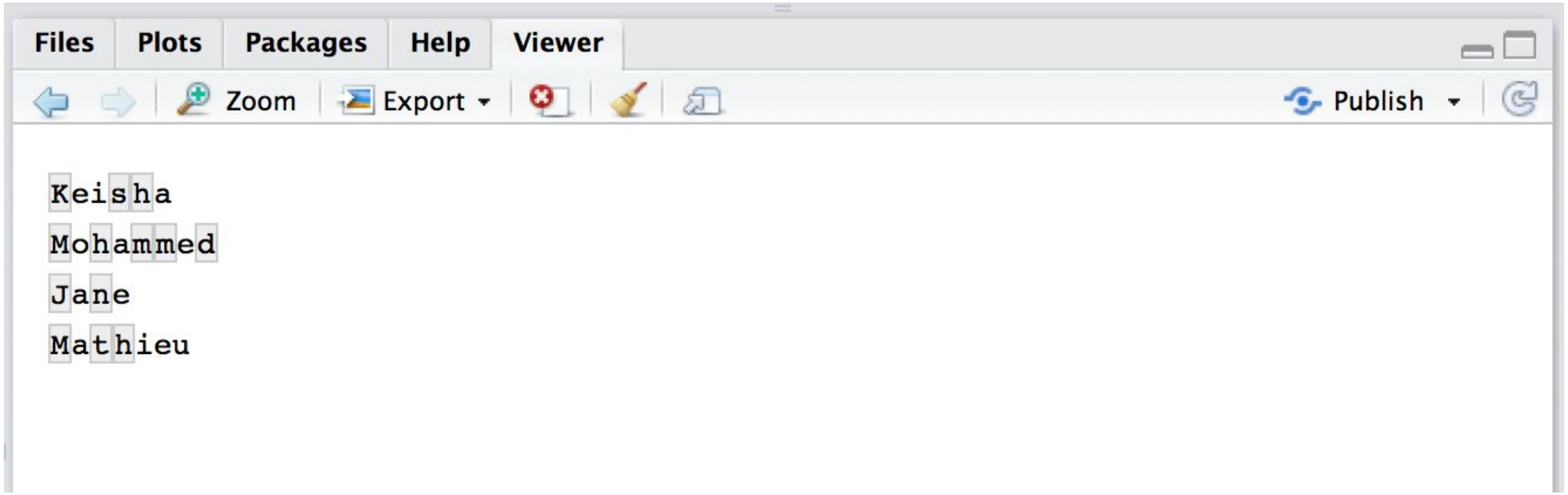
- "[aeiou]" : matches a, e, i, o, or u
- "[^aeiou]": matches anything *other than* a, e ,i , o, or u
- "\d" : matches any digit
- "\s" : matches any whitespace (space, tab, newline)
- "." : matches any character (except a newline)

```
names <- c("Keisha", "Mohammed", "Jane", "Mathieu")

## identify all lowercase vowels
str_view_all(names, "[aeiou]")
```



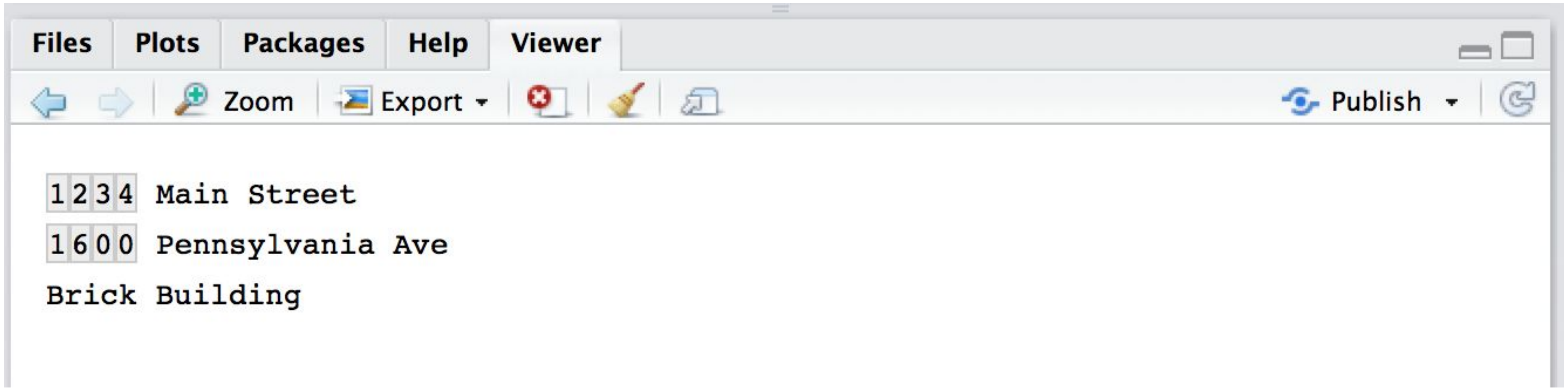
```
Keisha
Mohammed
Jane
Mathieu
```

```r
names <- c("Keisha", "Mohammed", "Jane", "Mathieu")

## identify anything that's NOT a lowercase vowel
str_view_all(names, "[^aeiou]")
```



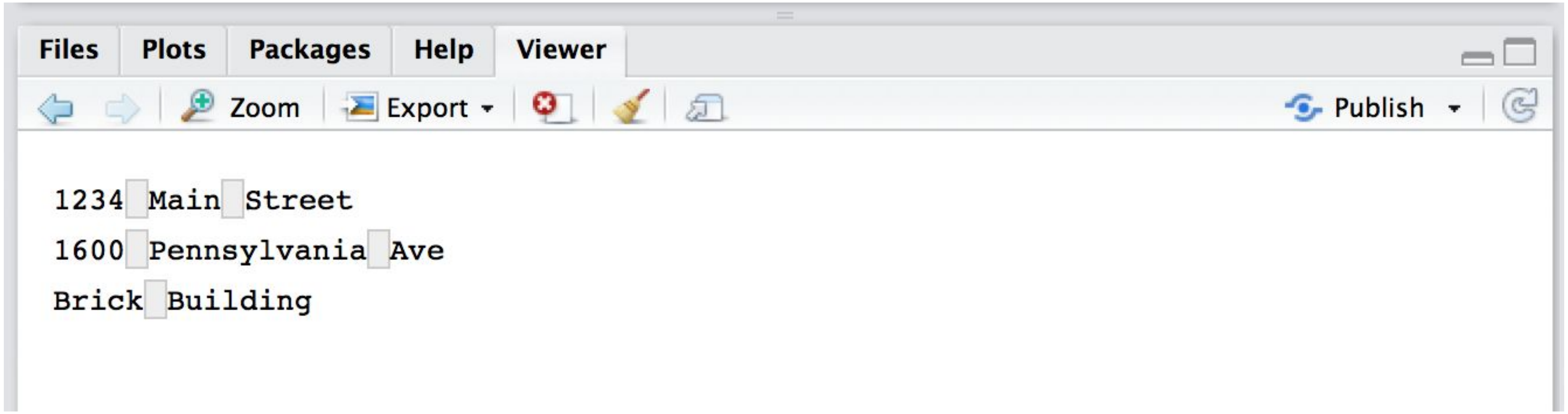
```
Keisha
Mohammed
Jane
Mathieu
```

```r
addresses <- c("1234 Main Street", "1600 Pennsylvania Ave", "Brick Building")

## identify anything that's a digit
str_view_all(addresses, "\\d")
```



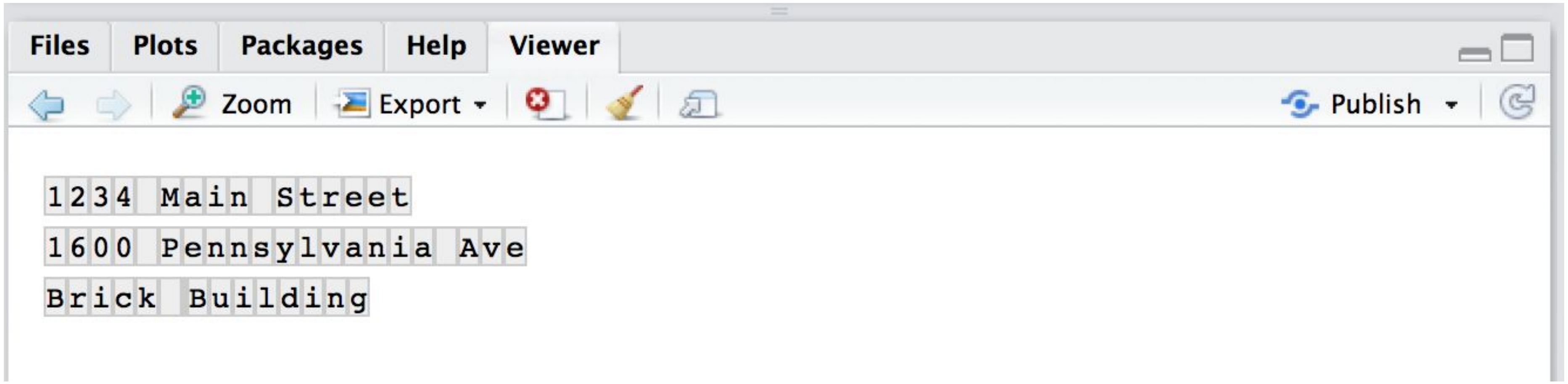
```
1234 Main Street
1600 Pennsylvania Ave
Brick Building
```

```
addresses <- c("1234 Main Street", "1600 Pennsylvania Ave", "Brick Building")

## identify any whitespace
str_view_all(addresses, "\\s")
```



```
1234 Main Street
1600 Pennsylvania Ave
Brick Building
```

```
addresses <- c("1234 Main Street", "1600 Pennsylvania Ave", "Brick Building")

## identify any character
str_view_all(addresses, ".")
```



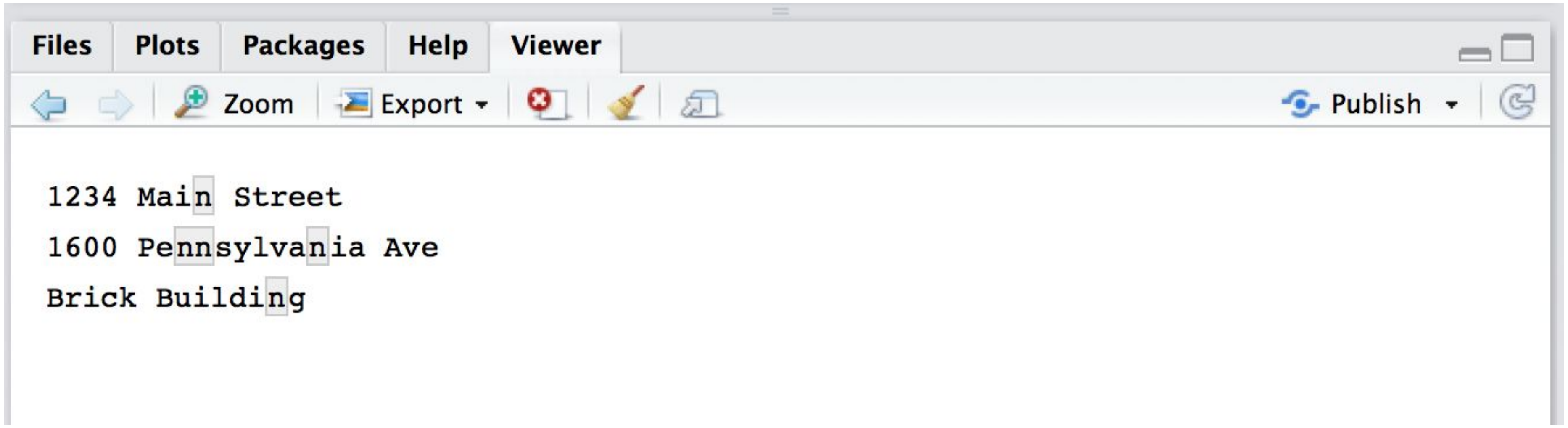
```
1234 Main Street
1600 Pennsylvania Ave
Brick Building
```

Repetition within regexs

- ? : 0 or 1
- + : 1 or more
- \\* : 0 or more

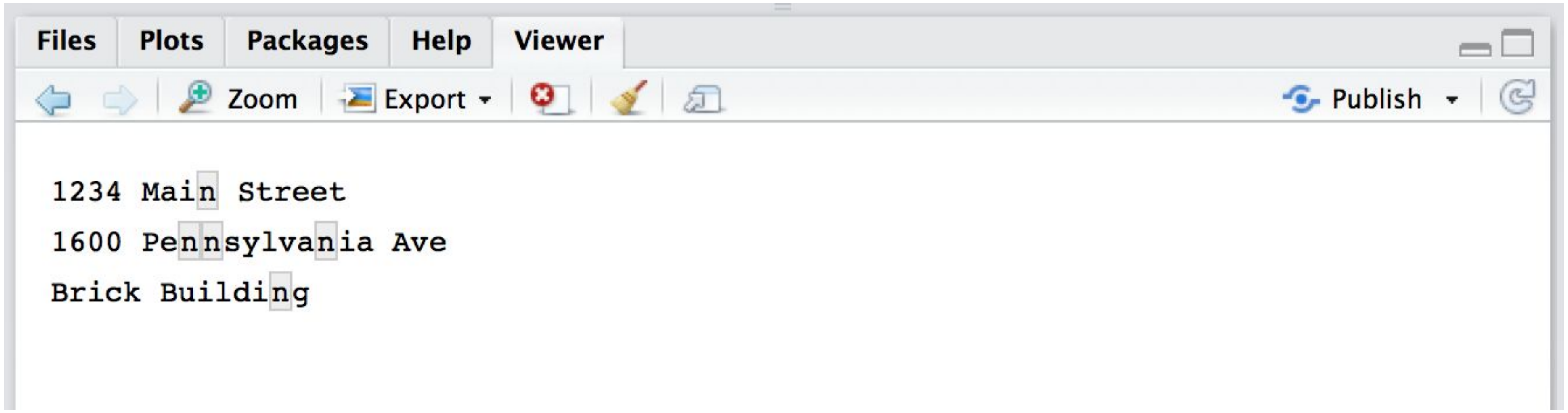
- {n} : exactly n times
- {n,} : n or more times
- {n,m} : between n and m times

```
addresses <- c("1234 Main Street", "1600 Pennsylvania Ave", "Brick Building")

## identify any time n shows up one or more times
str_view_all(addresses, "n+")
```



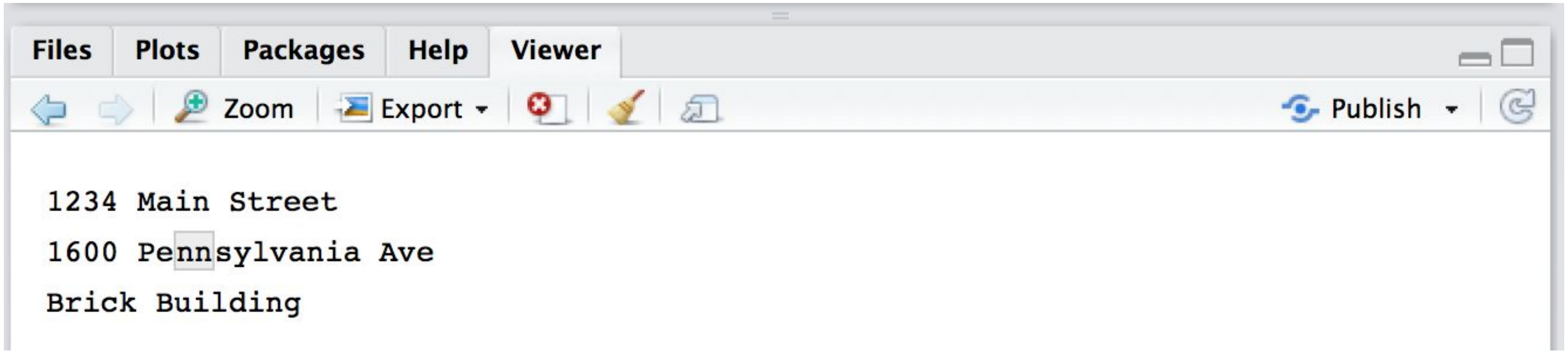
```
1234 Main Street
1600 Pennsylvania Ave
Brick Building
```

```r
addresses <- c("1234 Main Street", "1600 Pennsylvania Ave", "Brick Building")

## identify any time n shows up
str_view_all(addresses, "n{1}")
```



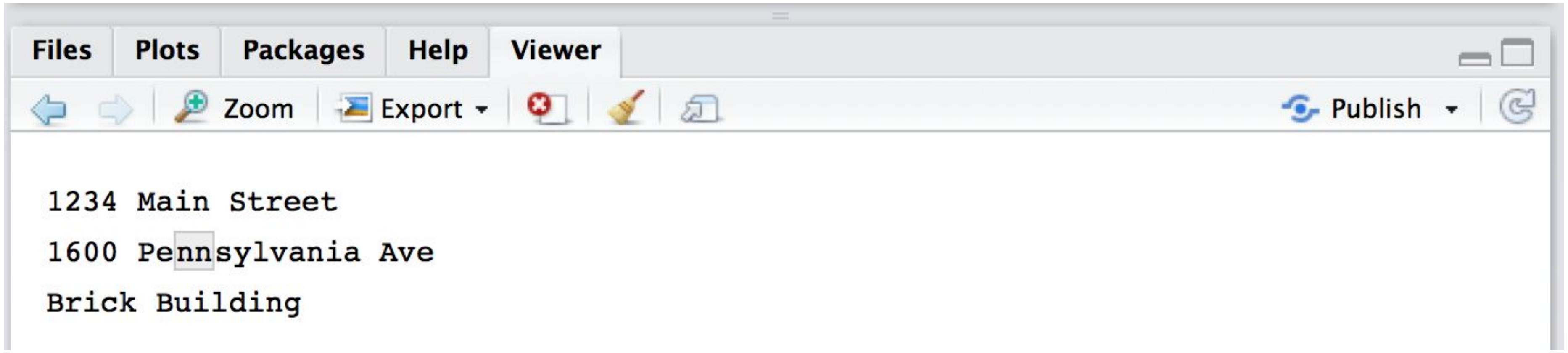
```
1234 Main Street
1600 Pennsylvania Ave
Brick Building
```

```
addresses <- c("1234 Main Street", "1600 Pennsylvania Ave", "Brick Building")

## identify any time n shows up exactly two times in a row
str_view_all(addresses, "n{2}")
```



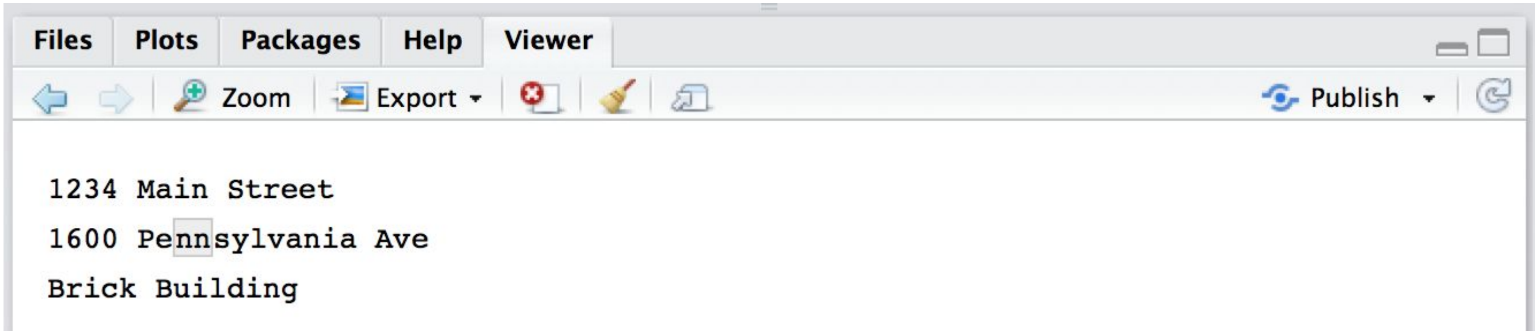
```
1234 Main Street
1600 Pennsylvania Ave
Brick Building
```

```r
addresses <- c("1234 Main Street", "1600 Pennsylvania Ave", "Brick Building")

## identify any time 'nn' shows up one or more times
str_view_all(addresses, "nn+")
```



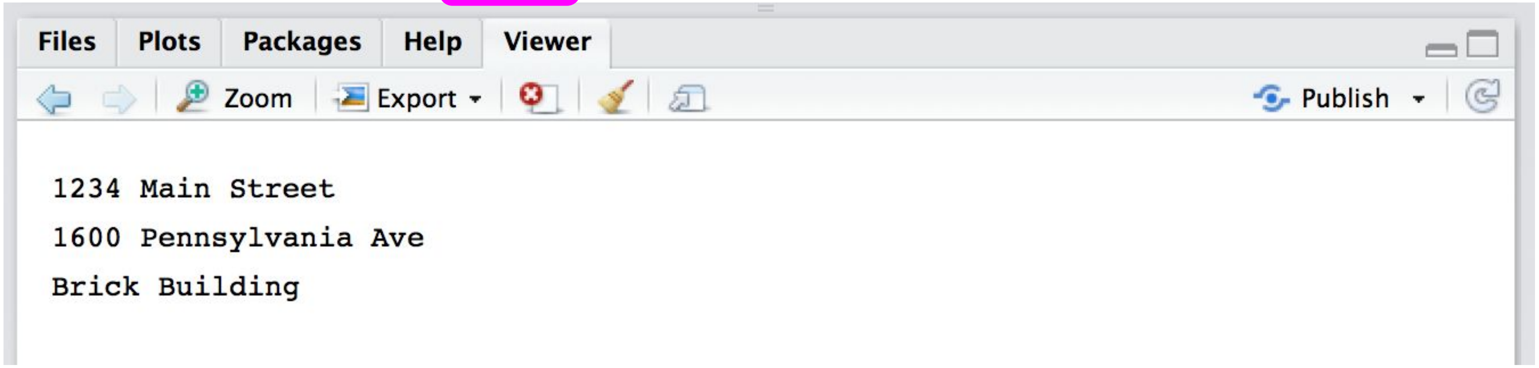
```
1234 Main Street
1600 Pennsylvania Ave
Brick Building
```

```r
addresses <- c("1234 Main Street", "1600 Pennsylvania Ave", "Brick Building")

## identify any time n shows up two or three times
str_view_all(addresses, "n{2,3}")
```



```
1234 Main Street
1600 Pennsylvania Ave
Brick Building
```

```r
## identify any time n shows up three or four times
str_view_all(addresses, "n{3,4}")
```



```
1234 Main Street
1600 Pennsylvania Ave
Brick Building
```

stringr

www.rstudio.com