

# All About Strings

This chapter introduces you to the basic concepts of character strings in R.

## We'll cover the following

- Creating Character Strings
  - Empty String
  - Escape Sequences
- Basic Operations with Strings
  - Length of a String
    - Why Do We Not Use length() Here?
  - Concatenating Two Strings
  - Duplicating Strings
    - Syntax for replicate()
    - Syntax for rep()

## Creating Character Strings #

In R, we can express character strings by surrounding text with double quotes:

“learning is fun”,

or we can also surround text with single quotes:

‘learning is fun’.

 Double\_Quotes

 Single\_Quotes

```
cat("learning is fun")
```



Creating string using double quotes

## Empty String #

The most basic type of string is the **empty** string produced by consecutive quotation marks: `""`.

`""` is a string with no characters in it, hence the name **empty string**:

Double\_Quotes

Single\_Quotes

cat("")

▶

📄

↶

⌂

Empty string using double quotes

Escape Sequences #

A sequence that starts with a `\` in a string is called an **escape sequence**. It allows us to include special characters in our strings.

Common escape sequences are:

Escape sequence	Usage
<code>\n</code>	newline
<code>\t</code>	tab
<code>\\</code>	backslash
<code>\'</code>	Single quote (')
<code>\"</code>	Double quote (")

You saw one escape sequence previously: `\n` is used to denote a new line.

Each escape sequence is considered one character.

 <code>'\n'</code>	 <code>'\t'</code>	 <code>'\\'</code>
---	---	---

```
cat("Learning\nis\nfun")
```



Using newline escape sequence





## Basic Operations with Strings #

The following are some of the basic methods for string manipulation. Have a look at their codes.

### Length of a String #

The length of a string can be found using a simple method `nchar()`. Let's find the length of the string "learning is fun".

```
nchar("learning is fun")
```







Example of how to use nchar()

Empty spaces or tabs, i.e. `\t`, are also considered characters and are added in the total length of the string.

### Why Do We Not Use `length()` Here? #

The keyword `length()` gives the length of **R objects** for which this method has been defined. It returns the number of **elements** in the object, so for a *single string* it will return 1.

```
length("learning is fun")
```



Example of how to use length()

## Concatenating Two Strings #

Two strings can be concatenated using `paste()`.

```
paste("learning", "is", "fun")
```

The image shows a snippet of R code in a light gray editor box: `paste("learning", "is", "fun")`. To the right of the code is a small copy icon. Below the editor is a toolbar with a blue play button on the left and three icons on the right: a blue save icon, a gray back arrow, and a gray full-screen icon.

Example of how to use `paste()`

By default, `paste()` inserts a single space between pairs of strings.

However, this default setting can be overridden using the `sep` argument in `paste()`.

For example, we can set a `.` or an empty string `""` as a separator.

 `sep="."`

 `sep=""`

```
paste("learning", "is", "fun", sep = ".")
```

The image shows an RStudio interface. At the top, there are two tabs: the first has an R logo and the text `sep="."`, the second has an R logo and the text `sep=""`. Below the tabs is a light gray editor box containing the code `paste("learning", "is", "fun", sep = ".")`. To the right of the code is a small copy icon. Below the editor is a toolbar with a blue play button on the left and three icons on the right: a blue save icon, a gray back arrow, and a gray full-screen icon.

`paste()` with `sep = "."` argument

## Duplicating Strings #

We can duplicate the same string multiple times using the keywords `replicate()` or `rep()`.

Syntax for `replicate()` #

```
replicate(<numberOfTimes>, <stringToReplicate>)
```

Syntax for `rep()` #

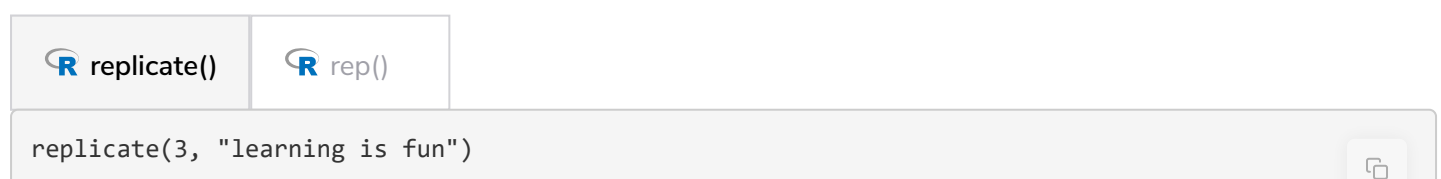
```
rep(<stringToReplicate>, <numberOfTimes>)
```

The parameter `numberOfTimes` specifies how many times to repeat the parameter `stringToReplicate`.

 `replicate()`

 `rep()`

```
replicate(3, "learning is fun")
```

The image shows an RStudio interface. At the top, there are two tabs: the first has an R logo and the text `replicate()`, the second has an R logo and the text `rep()`. Below the tabs is a light gray editor box containing the code `replicate(3, "learning is fun")`. To the right of the code is a small copy icon. Below the editor is a toolbar with a blue play button on the left and three icons on the right: a blue save icon, a gray back arrow, and a gray full-screen icon.



Example of how to use replicate()

Both `rep` and `replicate` perform the same task. However, the order of the arguments is the opposite.

---

In the next lesson we will be learning an important concept: the difference between `cat()` and `print()`. It is particularly important to understand this concept early on as we'll be moving on to more difficult concepts later in the course.