

## Set 27: String Operations

**Skill 27.01: Concatenate String type variables**

**Skill 27.02: Use .length to find the number of characters in a String**

**Skill 27.03: Retrieve a portion of a String using the substring() function**

**Skill 27.04: Convert between lower and upper case**

**Skill 27.05: Use escape sequences to print special characters**

**Skill 27.06: Retrieve a character at a specific index in a String**

**Skill 27.07: Trim excess white space before and after a String**

**Skill 27.08: Explore more String operations**

### Skill 27.01: Concatenate String type variables

#### Skill 27.01 Concepts

In JavaScript a String is defined as any set of characters stored in between single or double quotes. Strings can include letters, symbols, and numbers.

```
var someString = "This is a String!";  
var someString2 = 'Here is another String';  
var someString3 = "100";
```

Concatenation is the process of attaching Strings together. In JavaScript this is done with the plus (+) sign.

Consider the following example. Below, the variables initialized to “Hello” and “good buddy” are concatenated and assigned to a new variable c. The program prints Hellogood buddy to the screen.

```
var mm = "Hello";  
var nx = "good buddy";  
var c = mm + nx;  
console.log //prints Hellogood buddy... notice no space between o & g
```

A space between the words “Hello” and “good” could have been achieved by concatenating a space between these words as shown below,

```
var mm = "Hello";  
var nx = "good buddy";  
console.log(mm+ " " + nx); //prints Hello good buddy..notice the space
```

The example below also illustrates another way a space could have been achieved,

```
console.log("Hello" + " good buddy"); //prints Hello good buddy
```

It is also possible to concatenate a String with a numeric variable as follows. This is a useful technique for converting a number type variable to a String variable type.

```
var x = 17;
```

```
var s = "Was haben wir gemacht?"//German for "What have we done"
var combo = s + "" + x;
console.log(combo);//prints Was haben wir gemacht?17
```

### [Skill 27.01 Exercise 1](#)

### Skill 27.02: Use length to find the number of characters in a String

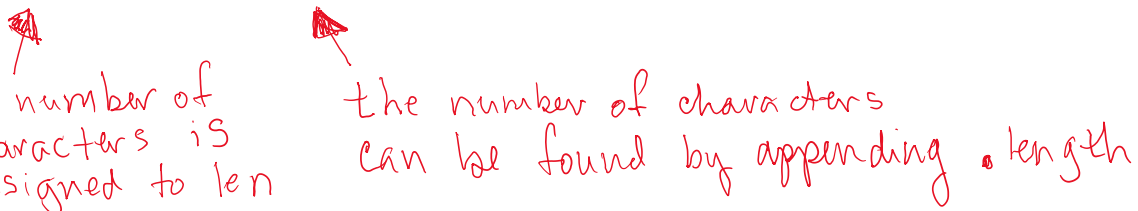
#### Skill 27.02 Concepts

While *String* type variables are nothing more than a series of characters, they are also considered objects in JavaScript (as opposed to primitives). The String object class in JavaScript provides a library of functions which are useful for manipulating *String* type variables.

To access these functions, we can use the "dot" notation. One function in the String library that is useful is the length function. This function can be used to find the number of characters in a String. Consider the following example,

#### Code

```
var theName = "Donald Duck";
var len = theName.length;
```



```
console.log(len);//prints 11, notice the space gets counted
```

### [Skill 27.02 Exercise 1](#)

### Skill 27.03: Retrieve a portion of a String using the substring() function

#### Skill 27.03 Concepts

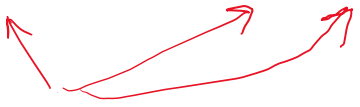
The substring() function can be used to indicate the portion of the String we want to retrieve. In computer science counting begins at 0. In the String "Sparky the dog", the character assignments are as follows,

| Position | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|----------|---|---|---|---|---|---|---|---|---|---|----|----|----|----|
| Letter   | S | p | a | r | k | y |   | t | h | e |    | d  | o  | g  |

If we wanted to print all the characters starting at position 4 ("k"), we could use substring(4). This is illustrated below,

#### Code

```
var myPet = "Sparky the dog";  
var smallPart = myPet.substring(4);
```



To retrieve a portion of a String, ".substring" is appended to the String. The number in parentheses indicates the character where the new String begins (at character 4).


```
console.log(smallPart); // prints ky the dog
```

Another application of the substring() function involves retrieving the characters from the middle of a String.

If we want to print all the characters starting at k (position 4) and ending at d (position 11), we could use substring(4, 12). This is illustrated below,

#### Code

```
var smallPart2 = myPet.substring(4,12);
```



The number 4 indicates the starting position  
the number 12 indicates the ending position.  
All the letters starting at position 4 and  
less than position 12 will be printed.

```
console.log(smallPart2); // prints ky the d
```

#### [Skill 27.03 Exercise 1](#)

## Skill 27.04: Convert between lower and upper case

### Skill 27.04 Concepts

The function “toLowerCase()” converts all characters to lower case (small letters)

#### Code

```
var phrase = "Where is my car?";  
var phraseLower = phrase.toLowerCase();
```

↖  
Appending .toLowerCase() to the String  
converts all the characters to lower case

```
console.log(phraseLower); //prints where is my car?
```

The function “toUpperCase()” converts all characters to upper case (capital letters)

#### Code

```
var phrase = "Where is my car?";  
var phraseUpper = phrase.toUpperCase();
```

↖  
Appending .toUpperCase() to the String  
converts all the characters to upper case

```
console.log(phraseUpper); //prints WHERE IS MY CAR?
```

### [Skill 27.04 Exercises 1 & 2](#)

## Skill 27.05: Use escape sequences to print special characters

### Skill 27.05 Concepts

The following code block results in an error because quotes are not allowed in quotes in JavaScript.

```
console.log("What is the "right" way?"); //Error!
```

To force a quote character (") to printout, or to be part of a String, use the escape sequence, “\”. Note escape sequences always start with a “\” character.

Consider the following example,

| Code  |
|---|
| <pre>console.log("What is the \"right\" way?");</pre> <p>The <code>"\"</code> symbol before the <code>"</code> allows the <code>"</code> to be printed (or escaped)</p> |
| Output  |
| <pre>What is the "right" way?</pre>   |

Another escape sequence, `\n`, will create a new line (also called a line break) as shown below

| Code   |
|--|
| <pre>var msg = "Here is one line \n and here is another";<br/>console.log(msg);</pre> <p><code>"\n"</code> separates the String into 2 lines</p> |
| Output   |
| <pre>Here is one line<br/>and here is another</pre>  |

The escape sequence, `\\`, will allow us to print a backslash within our String. Otherwise, if we try to insert just a single `\` it will be interpreted as the beginning of an escape sequence,

| Code   |
|--|
| <pre>var path = "C:\\SomeFile.pdf";</pre> <p><i>the \ symbol can be used to escape the backslash</i></p> <pre>console.log(path);</pre> |
| Output   |
| C:\SomeFile.pdf  |

The escape sequence, \t, will allow us to tab over. The following code illustrates this,

| Code  |
|---|
| <pre>var msg2 = "Everything\t\tis\t\ttabbed";</pre> <p><i>there are 2 tabs inserted between each word</i></p> <pre>console.log(msg2);</pre> |
| Output  |
| Everything        is        tabbed  |

#### [Skill 27.05 Exercise 1](#)

#### Skill 27.06: Retrieve a character a specific index in a String

##### Skill 27.06 Concepts

Recall the following position designations for the String, "Sparky the dog"

| Position | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|----------|---|---|---|---|---|---|---|---|---|---|----|----|----|----|
| Letter   | S | p | a | r | k | y |   | t | h | e |    | d  | o  | g  |

The charAt function can be applied to return a character at a specified index. Consider the following example,

| Code  |
|---|
| <pre>var somePhrase = "Sparky the dog"; var firstLetter = somePhrase.charAt(0); var lastLetter = somePhrase.charAt(somePhrase.length-1); //14 - 1 = position 13 console.log("The first letter is " + firstLetter + ". The last letter is " + lastLetter + ".");</pre> |
| Output  |
| The first letter is S. The last letter is g.  |

#### [Skill 27.06 Exercise 1](#)

#### Skill 27.07: Trim excess white space before and after a String

##### Skill 27.07 Concepts

The built-in trim function is useful for trimming off extra whitespace before and after a String, while leaving the interior whitespace intact. (Whitespace consists of new line (\n), tab (\t), and spaces)

| Code  |
|---|
| <pre>var someText = "Take a Hike!"; someText = "\t\t" + someText + "\n"; //add some space before and after console.log("X" + someText.trim() + "X"); //prints XTake a HikeX</pre> |
| Output  |
| XTake a Hike!X  |

#### [Skill 27.07 Exercise 1](#)

#### Skill 27.08: Explore more String operations

##### Skill 27.08 Concepts.

In addition to the String operations covered in this lesson, there are many more. Check out the link below to explore more,

[https://www.w3schools.com/jsref/jsref\\_obj\\_string.asp](https://www.w3schools.com/jsref/jsref_obj_string.asp)