**STRINGS**

1. Strings can be created as primitives, from string literals, or as objects, using the [String()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/String) constructor:

const string1 = "A string primitive";

const string2 = 'Also a string primitive';

const string3 = `Yet another string primitive`

const string4 = new String("A String object");

1. Note that JavaScript distinguishes between String objects and [primitive string](https://developer.mozilla.org/en-US/docs/Glossary/Primitive) values.

String literals (denoted by double/single quotes) and strings returned from String calls in a non- constructor context (that is, called without using the [new](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/new) keyword) are primitive strings. In contexts where a method is to be invoked on a primitive string or a property lookup occurs, JavaScript will automatically wrap the string primitive and call the method or perform the property lookup on the wrapper object Instead.

const strPrim = "foo"; // A literal is a string primitive

const strPrim2 = String(1); // Coerced into the string primitive "1"

const strPrim3 = String(true); // Coerced into the string primitive "true"

const strObj = new String(strPrim); // String with new returns a string wrapper object.

console.log(typeof strPrim); // "string"

console.log(typeof strPrim2); // "string"

console.log(typeof strPrim3); // "string"

console.log(typeof strObj); // "object"

1. String primitives and String objects also give different results when using [eval()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/eval). Primitives passed to eval are treated as source code; String objects are treated as all other objects are, by returning the object. For example:

const s1 = "2 + 2"; // creates a string primitive

const s2 = new String("2 + 2"); // creates a String object

console.log(eval(s1)); // returns the number 4

console.log(eval(s2)); // returns the string "2 + 2"

1. A String object can always be converted to its primitive counterpart with the [valueOf()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/valueOf) method.

console.log(eval(s2.valueOf())); // returns the number 4

1. **Methods**
   1. **At()**
      1. The at() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values takes an integer value and returns a new [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) consisting of the single UTF-16 code unit located at the specified offset.
      2. This method allows for positive and negative integers. Negative integers count back from the last string character.
      3. Returns [undefined](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/undefined) if the given index can not be found.
   2. **charAt()**
      1. The charAt() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values returns a new string consisting of the single UTF-16 code unit at the given index.
      2. **This method allows positive integers in range of 0 to string.length - 1.**
      3. **Undefined is treated as 0 index**
      4. If index is out of the range returns an empty string.
   3. **charCodeAt()**
      1. The charCodeAt() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values returns an integer between 0 and 65535 representing the UTF-16 code unit at the given index.
      2. Zero-based index of the character to be returned
      3. If index is out of range of 0 – str.length - 1, charCodeAt() returns [NaN](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/NaN).
   4. **Concat()**
      1. The concat() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values concatenates the string arguments to this string and returns a new string.
      2. Syntax

concat()

concat(str1, str2, /\* …, \*/ strN)

* + 1. **Returns a new string I.e does not modify original string**
  1. **endsWith()**
     1. The endsWith() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values determines whether a string ends with the characters of this string, returning true or false as appropriate.
     2. Synatx

endsWith(searchString)

endsWith(searchString, endPosition)

* + 1. **endsWith will check from 0 to endPosition index**
    2. Retuen true if the given characters are found at the end of the string, including when searchString is an empty string; otherwise, false.
  1. **startsWith()**
     1. The startsWith() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values determines whether this string begins with the characters of a specified string, returning true or false as appropriate.
     2. **Syntax**

**startsWith(searchString)**

**startsWith(searchString, position)**

* + 1. **Position is t**he start position at which searchString is expected to be found (the index of searchString's first character). Defaults to 0.
  1. **Includes()**
     1. The includes() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values performs a case-sensitive search to determine whether a given string may be found within this string, returning true or false as appropriate.
     2. **Syntax**

**includes(searchString)**

**includes(searchString, position)**

* + 1. The position within the string at which to begin searching for searchString. (Defaults to 0.)
    2. Return true if the search string is found anywhere within the given string, including when searchString is an empty string; otherwise, false.
  1. **indexOf()**
     1. The indexOf() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values searches this string and returns the index of the first occurrence of the specified substring. It takes an optional starting position and returns the first occurrence of the specified substring at an index greater than or equal to the specified number.
     2. Synatax

**indexOf(searchString)**

**indexOf(searchString, position)**

* + 1. **If position is given less than 0 then it will be treated as 0 only by JS**
    2. **Return index of firstOccur if found else -1**
    3. **Return value when using an empty search string is just the position which we passed,by default 0.**
  1. **lastIndexOf()**
     1. The lastIndexOf() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values searches this string and returns the index of the last occurrence of the specified substring. It takes an optional starting position and returns the last occurrence of the specified substring at an index less than or equal to the specified position.
     2. If position is greater than the length of the calling string, the method searches the entire string. If position is less than 0, the behavior is the same as for 0 — that is, the method looks for the specified substring only at index 0.
     3. Return The index of the last occurrence of searchString found, or -1 if not found
  2. **padEnd()**
     1. The padEnd() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values pads this string with a given string (repeated, if needed) so that the resulting string reaches a given length. The padding is applied from the end of this string.
     2. Synatx

**padEnd(targetLength)**

**padEnd(targetLength, padString)**

* + 1. If the value of targetLength is less than or equal to str.length, the current string will be returned as-is.
    2. . If padString is too long to stay within targetLength, it will be truncated from end or start depending upon the type of language.The default value for this parameter is " "
  1. **padStart()**
     1. The padStart() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values pads this string with another string (multiple times, if needed) until the resulting string reaches the given length. The padding is applied from the start of this string.
     2. Syntax

**padStart(targetLength)**

**padStart(targetLength, padString)**

* + 1. If the value of tatgetlength less than or equal to str.length, then str is returned as-is.
    2. If padString is too long to stay within the targetLength, it will be truncated from the end.The default value for this parameter is " "
  1. **Repeat()**
     1. The repeat() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values constructs and returns a new string which contains the specified number of copies of this string, concatenated together.
     2. Takes an integer between 0 and [+Infinity](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Number/POSITIVE_INFINITY), indicating the number of times to repeat the string.
     3. Returns a new string containing the specified number of copies of the given string.
  2. **Replace()**
     1. The replace() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values returns a new string with one, some, or all matches of a pattern replaced by a replacement. The pattern can be a string or a [RegExp](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/RegExp), and the replacement can be a string or a function called for each match. If pattern is a string, only the first occurrence will be replaced. The original string is left unchanged.
     2. **Syntax-:**

replace(pattern, replacement)

* 1. **replaceAll()**
     1. The replaceAll() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values returns a new string with all matches of a pattern replaced by a replacement. The pattern can be a string or a [RegExp](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/RegExp), and the replacement can be a string or a function to be called for each match. The original string is left unchanged.
  2. **Splice()**
     1. The slice() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values extracts a section of this string and returns it as a new string, without modifying the original string.
     2. Syntax

**slice(indexStart)**

**slice(indexStart, indexEnd)**

* + 1. **indexEnd is exclusive**
    2. If indexStart >= str.length, an empty string is returned.
    3. If indexStart < 0, the index is counted from the end of the string. More formally, in this case, the substring starts at max(indexStart + str.length, 0).
    4. If indexEnd < 0, the index is counted from the end of the string. More formally, in this case, the substring ends at max(indexEnd + str.length, 0).
    5. If indexEnd <= indexStart after normalizing negative values (i.e. indexEnd represents a character that's before indexStart), an empty string is returned.
  1. **split()**
     1. The split() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values takes a pattern and divides this string into an ordered list of substrings by searching for the pattern, puts these substrings into an array, and returns the array.
     2. Syntax

**split(separator)**

**split(separator, limit)**

* + 1. **Limit is a** non-negative integer specifying a limit on the number of substrings to be included in the array.
    2. **I**f limit is 0, [] is returned.
  1. **Substr()**
     1. The substr() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values returns a portion of this string, starting at the specified index and extending for a given number of characters afterwards.
     2. Syntax

**substr(start)**

**substr(start, length)**

* + 1. **Length is t**he number of characters to extract.
    2. Returns a new string of specified length
  1. **Substring()**
     1. The substring() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values returns the part of this string from the start index up to and excluding the end index, or to the end of the string if no end index is supplied.
     2. **Returns a** new string containing the specified part of the given string.
  2. toLowerCase()
     1. The toLowerCase() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values returns this string converted to lower case.
     2. Returns a new string
  3. toUpperCase()
     1. The toUpperCase() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values returns this string converted to uppercase.
     2. Returns a new string
  4. Trim()
     1. The trim() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values removes whitespace from both ends of this string and returns a new string, without modifying the original string.
  5. trimStart()
     1. The trimStart() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values removes whitespace from the beginning of this string and returns a new string, without modifying the original string. trimLeft() is an alias of this method.
  6. **trimEnd()**
     1. The trimEnd() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values removes whitespace from the end of this string and returns a new string, without modifying the original string. trimRight() is an alias of this method.
  7. valueOf()
     1. The valueOf() method of [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) values returns this string value.
     2. Returns a string representing the primitive value of a given [String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) object.\
     3. Example

**const stringObj = new String('foo');**

**console.log(stringObj);**

**// Expected output: String { "foo" }**

**console.log(stringObj.valueOf());**

**// Expected output: "foo"**