String Properties and Functions

In this lesson, we'll look at some string properties and some useful string functions.
Let's begin!

WE'LL COVER THE FOLLOWING

The `String` type has several specific instance methods and properties, as summarized in the table below:

String instance methods and properties

Method/Property	Description
`length`	Gets the length of the string
`charAt()`	Returns the character at the specified index from a string. Indexes start with 0.
`charCodeAt()`	Returns the numeric Unicode value of the character at the specified index.
`concat()`	Combines the text of two or more strings passed as arguments, and returns a new string.
`indexOf()`	Returns the index within the calling `String` object of the first occurrence of the specified search value (first argument), starting the search at an optionally specified index (second argument). If the search value is not found, -1 is returned.
`lastIndexOf()`	Returns the index within the calling `String` object of the last occurrence of the specified search value (first argument), or -1 if not found. The calling string is searched backward. You can specify an optional index (second argument) to start the search from.
`localeCompare()`	Returns a number indicating whether a reference string comes before or after or is the same as the given string in sort order. For more details, see the `localeCompare()` reference on MDN.
`match()`	Retrieves an object describing the matches when matching a string against a regular expression passed as the argument of the method.
`replace()`	Returns a new string with some or all matches of a pattern (first argument) replaced by a replacement (second argument). The pattern can be a string or a `RegExp`, and the replacement can be a string or a function to be called for each match.
`search()`	Executes the search for a match between a regular expression (first argument) and this `String` object.
`slice()`	Extracts a section of a string and returns a new string. The first argument is an integer value at which to begin the extraction. The second (optional) argument is an index at which to end the extraction. If the second index is negative, this index value should be added to the length of the string to calculate the end index of extraction.
`split()`	Splits a `String` object into an array of strings by separating the string into substrings. The method accepts a separator (first argument) that can be a string or regular expression. This separator specifies the character(s) to use for separating the string. An optional second argument, an integer, also can be passed to the method, which specifies a limit on the number of splits to be found.
`substr()`	Returns the characters in a string beginning at the specified location (first argument) through the specified number of characters (second argument). For more details, see the `substr()` reference on MDN.
`substring()`	Returns a subset of a string between one index (first argument) and another (second argument), or through the end of the string. For more details, see `substring()` reference on MDN.
`toLocalLowerCase()`	Returns the calling string value converted to lower case, according to any locale-specific case mappings.
`toLocalUpperCase()`	Returns the calling string value converted to upper case, according to any locale-specific case mappings.

`toLowerCase()`	Returns the calling string value converted to lowercase.
`toUpperCase()`	Returns the calling string value converted to uppercase.

Now that we've covered the `String` data type, let's proceed with the `Date` type in the *next lesson*.