STRING METHODS

|  |  |
| --- | --- |
| **1-String length**  The length property returns the length of a string:  let length = text.length;  **Output=”length of string”**  There are 3 methods for extracting a part of a string:   * slice(*start*, *end*) * substring(*start*, *end*) * substr(*start*, *length*)   **2-String slice()**  slice() extracts a part of a string and returns the extracted part in a new string.  The method takes 2 parameters: start position, and end position (end not included).  Slice out a portion of a string from position 7 to position 13:  let text = "Apple, Banana, Kiwi"; let part = text.slice(7, 13);  **Output=** **Banana**  If you omit the second parameter, the method will slice out the rest of the string:  let part = text.slice(7);  **Output= Banana, Kiwi**  If a parameter is negative, the position is counted from the end of the string: let part = text.slice(-12);  **Output= Banana, Kiwi**  This example slices out a portion of a string from position -12 to position -6:  let part = text.slice(-12, -6);  **Output= Banana**  **3-String substring()**  substring() is similar to slice().  The difference is that start and end values less than 0 are treated as 0 in substring().  Example  let str = "Apple, Banana, Kiwi"; let part = str.substring(7, 13);  **Output= Banana**  **4-String substr()**  substr() is similar to slice().  The difference is that the second parameter specifies the **length** of the extracted part.  Example  let str = "Apple, Banana, Kiwi"; let part = str.substr(7, 6);  **Output= Banana**  If you omit the second parameter, substr() will slice out the rest of the string.  let part = str.substr(7);  **Output= Banana, Kiwi**  If the first parameter is negative, the position counts from the end of the string.  let part = str.substr(-4);  **Output= Kiwi**  **5-String replace()**  The replace() method replaces a specified value with another value in a string:  Example  let text = "Please visit Microsoft!"; let newText = text.replace("Microsoft", "W3Schools");  **Output=** Please visit W3Schools  Note  The replace() method does not change the string it is called on.  The replace() method returns a new string.  The replace() method replaces **only the first** match  If you want to replace all matches, use a regular expression with the /g flag set. See examples below.  To replace case insensitive, use a **regular expression** with an /i flag (insensitive):  let text = "Please visit Microsoft!"; let newText = text.replace("MICROSOFT"/i, "W3Schools");  **Output=** Please visit W3Schools  To replace all matches, use a **regular expression** with a /g flag (global match):  Example  let text = "Please visit Microsoft and Microsoft!"; let newText = text.replace(/Microsoft/g, "W3Schools");  **Output=** Please visit W3Schools and W3Schools  **6-String replaceAll()**  The replaceAll() method allows you to specify a regular expression instead of a string to be replaced.  If the parameter is a regular expression, the global flag (g) must be set set, otherwise a TypeError is thrown. **Example** text = text.replaceAll(/Cats/g,"Dogs"); text = text.replaceAll(/cats/g,"dogs");  replaceAll() does not work in Internet Explorer.  **7-String toUpperCase()**  A string is converted to upper case with toUpperCase(): **Example** let text1 = "Hello World!"; let text2 = text1.toUpperCase();  **Output=HELLO WORLD!**  **8-String toLowerCase()**  A string is converted to lower case with toLowerCase():  let text1 = "Hello World!";         let text2 = text1.toLowerCase();  **Output=hello world!**  **9-String concat()**  concat() joins two or more strings: **Example** let text1 = "Hello"; let text2 = "World"; let text3 = text1.concat(" ", text2);  **Output = Hello World!**  The concat() method can be used instead of the plus operator. These two lines do the same: **Example** text = "Hello" + " " + "World!"; text = "Hello".concat(" ", "World!");  **Output = Hello World!**  **10-String trim()**  The trim() method removes whitespace from both sides of a string: **Example** let text1 = "      Hello World!      "; let text2 = text1.trim();  **Output = Hello World!**  **11-String trimStart()**  The trimStart() method works like trim(), but removes whitespace only from the start of a string. **Example** let text1 = "     Hello World!     "; let text2 = text1.trimStart();  **Output = Hello World!**  **12-String trimEnd()**  The trimEnd() method works like trim(), but removes whitespace only from the end of a string. **Example** let text1 = "     Hello World!     "; let text2 = text1.trimEnd();  **Output =      Hello World!**  **13-String padStart()**  The padStart() method pads a string with another string: **Example** let text = "5"; let padded = text.padStart(4,"x");  **Output = xxx5** **Note** The padStart() method is a string method.  To pad a number, convert the number to a string first.  See the example below. **Example** let numb = 5; let text = numb.toString(); let padded = text.padStart(4,"0");  **Output = 0005**  **14-String padEnd()**  The padEnd() method pads a string with another string: **Example** let text = "5"; let padded = text.padEnd(4,"x");  **Output = 5xxx**  **15-String charAt()**  The charAt() method returns the character at a specified index (position) in a string: **Example** let text = "HELLO WORLD"; let char = text.charAt(0);  **Output = “H”**  **16-String charCodeAt()**  The charCodeAt() method returns the unicode of the character at a specified index in a string:  The method returns a UTF-16 code (an integer between 0 and 65535). **Example** let text = "HELLO WORLD"; let char = text.charCodeAt(0);  **Output = 72**  **17-String split()** |  |
|  |  |

A string can be converted to an array with the split() method:

### **Example**

text.split(",")    // Split on commas  
text.split(" ")    // Split on spaces  
text.split("|")    // Split on pipe