

Язык JavaScript (основы:
строки)

JS

СТРОКИ

Строки полезны для хранения данных, которые можно представить в текстовой форме.

Наиболее частые операции со строками:

- ❖ проверка их длины,
- ❖ построение строки с помощью операций строковой конкатенации + и +=,
- ❖ проверка на существование или местоположение подстрок,
- ❖ извлечение подстрок с помощью метода `substring()` и тд.

Внутренним форматом строк, вне зависимости от кодировки страницы, является Юникод (Unicode).

Содержимое строки в JavaScript нельзя изменять, можно лишь создать новую строку на основании старой.

СТРОКИ

Для **создания строковой переменной** используются двойные или одинарные кавычки.

```
let str = "Строковая переменная";
```

```
let str2 = 'Другая строка';
```

В строках могут быть использованы специальные символы:

- ❖ перенос строки \n
 - ❖ табуляция \t
 - ❖ \uNNNN символ в кодировке Юникод с шестнадцатеричным кодом `NNNN`. Например, \u00A9 и др
- Для отображения специальных символов** в строке, их нужно **экранировать** - поставить обратный слеш перед символом, например, \\n

Если строка в одинарных (двойных) кавычках, то внутренние одинарные (двойные) кавычки тоже должны экранироваться:

```
let str = "Строковая \"Строка в кавычках\" переменная";
```

МЕТОДЫ И СВОЙСТВА СТРОК

Свойство **length** - получение длины строки.

Доступ к символам (нумерация начинается с 0):

- 1) метод **charAt(индекс)** - получение символа по индексу если символа нет, метод вернет пустую строку
- 2) квадратные скобки:

```
let str = "Hello";
```

```
str[0]; // "H" если символа нет, результат будет равен undefined
```

Смена регистра:

- 1) метод **toLowerCase()** меняет регистр на нижний
- 2) метод **toUpperCase()** меняет регистр на верхний

МЕТОДЫ И СВОЙСТВА СТРОК

Взятие подстроки: `substring`, `substr`, `slice`

Метод **`substring(indexA[, indexB])`** - возвращает подстроку с позиции **`indexA`** до, но не включая **`indexB`**.

- ❖ Если **`indexA`** равен **`indexB`**, метод **`substring()`** вернёт пустую строку;
- ❖ Если аргумент **`indexB`** опущен, метод извлечёт символы до конца строки;
- ❖ Если любой из аргументов меньше, либо равен нулю или равен NaN, он будет трактоваться как равный 0;
- ❖ Если любой из аргументов больше, чем длина строки, он будет трактоваться как равный длине строки
- ❖ Если **`indexA`** будет больше **`indexB`**, то метод сработает так, как если бы аргументы были поменяны местами;

МЕТОДЫ И СВОЙСТВА СТРОК

Метод **substr**(**start**[, **length**]) - возвращает из строки указанное количество символов (**length**), начиная с позиции (**start**).

- ❖ если **start** - отрицательное число, оно трактуется как (длина_строки - start);
- ❖ если **start** больше, либо равен длине строки, метод вернёт пустую строку;
- ❖ если **start** - отрицательное число, метод использует его как индекс символа, начиная с конца строки.
- ❖ если **start** - отрицательное число и по модулю больше длины строки, метод будет использовать 0 в качестве начального индекса;
- ❖ если **length** равен нулю или отрицателен, метод **substr()** вернёт пустую строку; если **length** опущен, метод **substr()** извлечёт все символы до конца строки.

МЕТОДЫ И СВОЙСТВА СТРОК

Метод **slice**(beginSlice[, endSlice]) - возвращает часть строки от beginSlice до, но не включая, endSlice. Смысл параметров как в substring.

- ❖ если beginSlice отрицателен, то трактуется как (длина_строки - beginSlice)
- ❖ если endSlice опущен, slice() извлечёт всё до конца строки.
- ❖ если endSlice отрицателен, то трактуется как (длина_строки - endSlice)

МЕТОДЫ И СВОЙСТВА СТРОК

Метод **includes**(**searchString**[, **position**]) - проверяет, содержит ли строка заданную подстроку (**searchString**), и возвращает, **true** или **false**.

Поиск начинается с начала строки, если не задана позиция в строке (**position**), с которой нужно начинать поиск строки.

Метод **includes**() является регистрозависимым

Метод **endsWith**(**s**) – возвращает **true**, если строка заканчивается подстрокой **s**.

Метод **startsWith**(**s**) – возвращает **true**, если строка начинается со строки **s**.

Метод **repeat**(**times**) – повторяет строку **times** раз.

СРАВНЕНИЕ СТРОК

Все строки имеют внутреннюю кодировку Юникод. Внутри JavaScript-интерпретатора все строки приводятся к единому виду. Каждому символу соответствует свой код.

При сравнении с помощью операторов сравнения символы сравниваются не по алфавиту, а по коду.

`String.fromCharCode(code)` - возвращает символ по коду `code`
`str.charCodeAt(pos)` - возвращает код символа на позиции `pos`

Метод `str1.localeCompare(str2)` - возвращает:

- ❖ -1, если `str1 < str2`,
- ❖ 1, если `str1 > str2`
- ❖ 0, если они равны.

[Подробнее о localeCompare](#)
[Интернационализация в JS](#)

СТРОКИ - ШАБЛОНЫ

Обратные кавычки `` используются для:

❖ переноса строки

Например,

```
console.log(`строка  
в обратных кавычках`);
```

❖ интерполяции - вставки выражений при помощи \${...}

Например,

```
let height = 100;
```

```
let width = 20;
```

```
console.log(`высота равна ${height},  
ширина равна ${width},  
площадь равна ${height * width}`);
```

ЧТО ПОЧИТАТЬ ДОПОЛНИТЕЛЬНО

<https://learn.javascript.ru/string> - Строки