









Frontender[1.0] JavaScript - Strings. Строки и их методы

 YouTube	https://youtu.be/os9PGNAEDYU
 Telegram	https://t.me/Dmitry_Kolotilshikov
 Github	https://github.com/DmitryKolotilshikov/
 Boosty	https://boosty.to/dmitry_ko
# Номер урока	13

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

https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/String

 Задачи к этому уроку тут https://boosty.to/dmitry_ko

Кроме регулярных печатных символов можно использовать специальные символы, которые можно закодировать, используя нотацию escape-последовательностей:

Код	Вывод
<code>\0</code>	нулевой символ (символ NUL)
<code>\'</code>	одинарная кавычка
<code>\"</code>	двойная кавычка
<code>\\</code>	обратный слеш
<code>\n</code>	новая строка
<code>\r</code>	возврат каретки
<code>\v</code>	вертикальная табуляция
<code>\t</code>	табуляция
<code>\b</code>	забой
<code>\f</code>	подача страницы
<code>\uXXXX</code>	Юникод-символ
<code>\xXX</code>	символ в кодировке Latin-1

Техническая информация Свойства **Кодировка** Отображение

String.fromCharCode(128077)

Кодировка	hex	dec (bytes)	dec	binary
UTF-8	F0 9F 91 8D	240 159 145 141	4036989325	11110000 10011111 10010001 10001101
UTF-16BE	D8 3D DC 4D	216 61 220 77	3627932749	11011000 00111101 11011100 01001101
UTF-16LE	3D D8 4D DC	61 216 77 220	1037585884	00111101 11011000 01001101 11011100
UTF-32BE	00 01 F4 4D	0 1 244 77	128077	00000000 00000001 11110100 01001101
UTF-32LE	4D F4 01 00	77 244 1 0	1307836672	01001101 11110100 00000001 00000000

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <script src="script.js"></script>
  <title>Strings</title>
</head>
<body>
</body>
</html>
```

```
// https://symb1.cc/ru/1F44D/ - Юникод (палец вверх)
// Strings (строки)
```

```
const log = console.log;
```

```

let str = "Hello \nWorld!";
str = "Hello \"man\"";
str = '👍';
str = String.fromCharCode(128077);

// str = "\u+1F44D"; // error
str = "\u{1F44D}";

const copyRight = "\u00A9";

// log(copyRight)

/*
Фигурные скобки для Unicode-символов выше U+FFFF: Unicode-символы выше U+FFFF (которые требуют более 4 шестнадцатеричных цифр) должны заключаться в фигурные скобки ({}) при использовании escape-последовательности \u.

Некорректный синтаксис: Синтаксис \u+1F44D неверен, потому что + не является частью синтаксиса escape-последовательностей в JavaScript. */

// -----

// const filePath = String.raw`C:\Development\profile\aboutme.html`;
// log(`The file was uploaded from: ${filePath}`);

// -----
const firstName = "Anna";

str = `Hello ${firstName}`;

str = `User:
- Alex
- Nik
- Julia
- Margarita
`;

// Ошибка с ""
// str = "User:
// - Alex
// - Nik
// - Julia
// - Margarita
// ";

// log(str);
// -----

let text = "World";

// log(text.length);
// log(text.toUpperCase());
// log(text.toLowerCase());
// log(text.repeat(3))

// log(text[0]);
// log(text[1]);

```

```

// log(text[4]);

// log(text.at(3));
// log(text.at(-1));
// log(text.at(-2));

let me = "I am " + text.at(1) + text.at(3) + text.at(4);

me = `I am ${text.at(1)}${text.at(3)}${text.at(4)}`;

// log(str);

// -----

str = "      JavaScript is awesome      ";
// str = str.trimStart();
// str = str.trimEnd();
// str = str.trim();

// log(str);
// log(str.replace("Script", ""));
// log(str.replace("Java", "HTML"));
// log(str.indexOf("S"));
// log(str.indexOf("Script"));
// log(str.indexOf("Java"));
// log(str.indexOf("W")); // -1 значит не найдено

str = "JavaScript is awesome";

if (str.indexOf("Java")) { // вернет 0, а 0 -это falsy значение
  console.log("проверка пройдена");
}
if (str.indexOf("Java") !== -1) { // 0 !== -1 это true
  console.log("проверка пройдена");
}

// -----

// log(str);
// log(str.includes("is"));
// log(str.startsWith("Java"));
// log(str.startsWith("is"));
// log(str.endsWith("some"));
// log(str.endsWith("is"));

// -----
// substring, slice, substr

// substring(start, end)
// slice(start, end)
// substr(start, length)

str = "hardcore";

// log(str.substring(0, 4)) // "hard"
// log(str.substring(4)) // "core"
// log(str.substring(8, 4)) // "core", если start больше end, то substring меняет их местами

```

```
// log(str.slice(0, 4)); // "hard"
// log(str.slice(8, 4)); // ""

// log(str.substr(1,3)) // deprecated - устарело

// -----

let random = "abc";

// log(random.padStart("10", 0));
// log(random.padEnd("10", 0));
// log(random.padEnd("13", Math.random()));
```