

Язык JavaScript (основы:
регулярные выражения)

JS

КАК ЗАДАТЬ ШАБЛОН

Регулярные выражения – мощное средство поиска и замены в строке. Регулярное выражение, состоит из паттерна (он же «шаблон») и необязательных флагов.

Синтаксис выражения:

```
let regexp = new RegExp(" ", " "); //шаблон, флаги
```

Шаблон - это строка, которая состоит из специальных символов.

Флаги:

i - если этот флаг есть, то регэксп ищет независимо от регистра, то есть не различает между А и а.

g - если этот флаг есть, то регэксп ищет все совпадения иначе **только первое**.

m – многострочный режим.

s - включает режим «dotall», при котором точка . может соответствовать символу перевода строки \n (ES2018).

u - включает полную поддержку юникода. Флаг разрешает корректную обработку суррогатных пар (ES2018).

y - режим поиска на конкретной позиции в тексте (ES2015).

КАК ЗАДАТЬ ШАБЛОН

Как правило, используют более короткую запись: шаблон внутри слешей "/".

```
let regexp = / /; // шаблон без флагов
```

```
let regexp = / /gmi; // gmi шаблон с флагами
```

Методы для поиска по регулярным выражениям встроены прямо в обычные строки String: `str.search(reg)` - возвращает позицию первого совпадения или `-1`, если ничего не найдено.

Пример:

```
let str = "Я люблю JavaScript!";
```

```
let regexp = /лю/;
```

```
alert(str.search(regexp) ); // 2
```

МЕТОД STR.MATCH(REG)

Метод `str.match` работает по-разному, в зависимости от наличия или отсутствия флага `g`, поэтому сначала мы разберём вариант, когда его нет. В этом случае `str.match(reg)` находит только одно, первое совпадение.

Результат вызова — это массив, состоящий из этого совпадения, с дополнительными свойствами `index` — позиция, на которой оно обнаружено и `input` — строка, в которой был поиск.

Пример:

```
let str = "ой-ой- ой";  
let result = str.match(/ой/i);  
alert(result[0]); // ой (совпадение)  
alert(result.index); // 0 (позиция)
```

МЕТОД STR.MATCH(REG)

Метод `str.match` при наличии флага `g` находит все совпадения.

Результат вызова — это массив, состоящий из найденных элементов.

Пример:

```
let str = "ой-ой- ой";  
let result = str.match(/ой/ig);  
alert(result[0]); // ой (совпадение)  
alert(result[1]); // ой (совпадение)  
alert(result[2]); // ой (совпадение)  
alert(result); // ["ой", "ой", "ой"]
```

МЕТОД REGEXP.TEST(STR)

Метод **test** проверяет, есть ли хоть одно совпадение в строке `str`. Возвращает **true/false**.

```
let str = "Люблю регэкспы я, но странною любовью";
```

```
let regexp = /лю/i;
```

```
// эти две проверки идентичны
```

```
alert(regexp.test(str)); // true
```

```
alert(/лю/i.test(str)); // true
```

КЛАСС СИМВОЛОВ ТОЧКА "."

Особым классом символов является точка ".", который соответствует любому символу, если с флагом регулярного выражения s, в противном случае любой символ, кроме перевода строки \n.

```
let re = /CS.4/;  
alert("CSS4".match(re)); // найдено "CSS4"  
alert("CS-4".match(re)); // найдено "CS4"  
alert("CS 4".match(re)); // найдено "CS 4"
```

КЛАССЫ И СПЕЦСИМВОЛЫ

`[a-z][abc....xyz]` — символьным классом описывается один символ в условии поиска

`[a-zA-Z]` — символьный класс со всеми буквами

латинского алфавита

`[^a-z]` — все кроме латинских букв

`[a-fq-x]` — два диапазона от а до f и q до x

`\w` — то же самое, что `[a-zA-Z0-9_]`,

`\s` Пробельный символ, включая табы, переводы строки и т.п.

КЛАССЫ

`[0-9]` — символьным классом описывается одна цифра в условии поиска

`\d` — Цифра, символ от 0 до 9.

`\D` — обратный класс, нецифра, то есть любой символ кроме `\d`, например буква.

КВАНТИФИКАТОРЫ

{2} — квантификатор , что выражает количество символов в условии поиска.

{3,5} — количество от 3 до 5 символов

? или **{0, 1}** — предыдущий символ является не обязательным

Применение квантификторов к классу

[az]{0,1} или **[az]?**

[az]* может идти сколь угодно

+ или **{1,0}** минимум 1-ин символ

ЭКРАНИРОВАНИЕ СПЕЦИАЛЬНЫХ СИМВОЛОВ

В регулярных выражениях есть и другие символы, имеющие особый смысл.

Они используются, чтобы расширить возможности поиска.

Вот их полный список : [\ ^ \$. | ? * + ().

Для экранирования используется обратный слэш '\'.

Пример:

```
let str = 'function() {alert(100)}';
```

```
let ptrn = /\(/;
```

```
str.match(ptrn);
```

ЗАДАЧИ

1. Определить что номер формата:
+7 (921) 911-51-97 введен корректно.
2. Определить что почта
info@ifmo.ru введена корректно.
3. Определить что дата 31.12.2015 введена
корректно.

ЧТО ПОЧИТАТЬ ДОПОЛНИТЕЛЬНО

<https://learn.javascript.ru/regular-expressions> - Раздел онлайн учебника посвященного Регулярным выражениям