РЕГУЛЯРНЫЕ ВЫРАЖЕНИЯ

```
// /pattern/flags
// new RegExp(pattern[, flags])
```

```
let str = "This is a javascript course. Lesson #6.";
let re = /Thi/g;
// далее мы будем использовать для примеров только
// литерал регулярного выражения.
```

```
// flags
g - global
i - ignore case
m - multiline
/JAVA/gim
```

```
// объединение в "классы"
/[jvscrpt]/g
```

```
// диапазоны
/[a-zA-T]/g
/[A-z]/g
/[0-9]/g
// исключающие диапазоны
/[^a-z]/g
```

```
// короткие записи частоиспользуемых классов
//d/g // === /[0-9]/g
//D/g // === /[^0-9]/g
//w/g // alphanumeric
//W/g // non-alphanumeric
```

```
/\b/ // границы слов
/\bis/ // starts with "is"
/se\b/ // ends with "se"
/\Bis/ // doesn't start with "is"
/es\B/ // doesn't end with "es"
```

```
// логическое "или"
/\bjava|script\b/ // starts with "java" OR ends with "script"
/\Bis|is\B/
```

```
let name = "Yuriy";
/(Y|I)uri(y|i)/g
/[YI]uri[yi]/g
```

```
/*
    /(https?:)?
    \/\/
    (\w+\.?\w+(?=\.\w+\/)?)
    (.\w+)
    (\/?.+?(?=\?|#))?
    \/??
    (.+(?=#))
    *#?
    (\w+)?/g
*/
```

```
// prevent grouping
let str = "Yuriy";
/(?:Y|I)uri(?:i|y)/g
```

```
/*
  /pattern/flags
  new RegExp(pattern[, flags])
*/
let re = /\d+/g,
  re2 = new RegExp(/\d+/, "g"),
  re3 = new RegExp("\d+", "g");
```



```
let str = "A string w/ strange spaces",
    re = /\s+/g;
str.split(re);
```

REGEXP.PROTOTYPE PROPS AND METHODS

```
let str = "this is {{handlebars}} template",
    noTpl = "simple string",
    re = /\{\{.+\}\}/;

re.test(str);
re.test(noTpl);
```

```
let str = "00 11 22 33 44 55",
    reg = /\d+/g;

reg.exec(str);
reg.lastIntex;

reg.exec(str);
reg.exec(str);
reg.exec(str);
reg.exec(str);
```

```
let str = "root@127.0.0.1",
    reg = /(\w+)@(.+)/g,
    match;

while(match = reg.exec(str)) {
    console.log(`
    substring: ${match[0]};
    username: ${match[1]};
        ip: ${match[2]}`);
}
```

ПОЛЕЗНЫЕ ССЫЛКИ

- Regexper (diagrams)
- Regexr (sandbox)
- RegExp (MDN)
- Регулярные выражения (скринкаст, ч.1)
- Регулярные выражения (скринкаст, ч.2)