

- Проверка пользовательского ввода с помощью JavaScript и регулярных выражений
- Обработка нескольких значений с использованием коллекций JavaScript
- Управление датами с помощью JavaScript Date API
- Практика: Создание приложения “meal-divider”
- Практика: Расчет общего количества в зависимости от возраста

Регулярные выражения



```
var regexp = /Java/;  
var regexp = /Java/gmi;  
var regexp = new RegExp('Java', 'gmi');
```

```
'I like JavaScript'.search(/Java/); // 7
```

```
'I like JavaScript'.search(/java/); // -1
```

```
'I like JavaScript'.search(/java/i); // 7
```

```
'ОЙ-Ой-ой'.match(/ой/i);  
// ['ОЙ', index: 0, input: 'ОЙ-Ой-ой']
```

```
'ОЙ-Ой-ой'.match(/ой/ig);  
// ['ОЙ', 'Ой', 'ой']
```

```
'I like JavaScript'.match(/java(script)/i);  
// ["JavaScript", "Script"]
```

```
'I like JavaScript'.match(/love/gi); //null
```

Регулярные выражения



```
'12-34-56'.replace(/-/,':');    // '12:34-56'  
'12-34-56'.replace(/-/g, ':');  // '12:34:56'
```

```
"Василий Пупкин".  
  replace(/(Василий) (Пупкин)/, '$2, $1');  
// Пупкин, Василий
```

```
var i = 0;  
  
"ОЙ-Ой-ой".replace(  
  /ой/gi,  
  function() {return ++i;}  
);  
  
// "1-2-3"
```

Регулярные выражения



```
var str="Читайте про JavaScript на javascript.ru";
var regexp=/javascript/ig;

while (result=regexp.exec(str)){
    console.log(result);
}

// ["JavaScript", index: 12, input: "Читайте про
JavaScript на javascript.ru"]

//["javascript", index: 26, input: "Читайте про
JavaScript на javascript.ru"]
```

```
/@/ig.test("me@mydomain.com"); //true
```

Регулярные выражения

<code>^</code>	начало текста	<code>?</code>	0 или 1
<code>\$</code>	конец текста	<code>*</code>	от 1 до ∞
<code>\b</code>	граница слова	<code>+</code>	от 1 до ∞
<code>\s</code>	пробельный символ	<code>{1-5}</code>	от 1 до 5
<code>\S</code>	не <code>\s</code>	<code>{3}</code>	ровно 3
<code>\w</code>	буква или <code>_</code>	<code>{2,}</code>	от 2 до ∞
<code>\W</code>	не <code>\w</code>	<code> </code>	или (альтернатива)
<code>\d</code>	цифра	<code>()</code>	подшаблон
<code>\D</code>	не цифра	<code>\</code>	экранирование
<code>.</code>	любой символ	<code>/m</code>	многострочный
<code>[abc]</code>	набор символов		
<code>[a-z]</code>	диапазон		
<code>[^abc]</code>	все, кроме abc		

Цвет

```
/#[0-9a-f]{6}/gi.test("#0dff97");
```

Форма

```
/<form(\s|>).*</form>/i
```

Число с дробной частью

```
/^\d+\.\d+$/ig
```

Map, Set, WeakMap, WeakSet (ES6)



```
let s = new Set()
s.add("hello").add("goodbye").add("hello")
s.size === 2
s.has("hello") === true
for (let key of s.values()) console.log(key)
```

```
let m = new Map()
m.set("key1", 42)
m.set("key2", 34)
m.get("key2") === 34
m.size === 2
for (let [ key, val ] of m.entries())
  console.log(key + " = " + val)
```

WeakMap, WeakSet – не хранят указатели

Date

`new Date()`

`new Date(milliseconds)`

`new Date(dateString)`

`new Date(year, month, day, hours, minutes, seconds, milliseconds)`

```
new Date ( )
```

```
new Date (864000000)
```

```
new Date ("2017-06-15") //ISO
```

```
new Date ("2017-06") //ISO
```

```
new Date ("2017") //ISO
```

```
new Date ("2017-06-15T11:35:00Z") //ISO UTC
```

```
new Date ("2017-06-15T11:35:00+03:00") //ISO +3
```

```
new Date (2017, 5, 15, 11, 33, 30, 0)
```

```
new Date (2017, 5, 15)
```



```
<p id="demo"></p>

<script>
  d = new Date();
  document.getElementById("demo").innerHTML = d;
  // toString()
</script>
```

toString ()

toUTCString ()

toDateString ()

Date

getDate()	getUTCDate()	setDate()
getDay()	getUTCDay()	
getFullYear()	getUTCFullYear()	setFullYear()
getHours()	getUTCHours()	setHours()
getMilliseconds()	getUTCMilliseconds()	setMilliseconds()
getMinutes()	getUTCMinutes()	setMinutes()
getMonth()	getUTCMonth()	setMonth()
getSeconds()	getUTCSeconds()	setSeconds()
getTime()		setTime()

```
new Date().setFullYear(2015) < new Date() //true
```

- Проверка пользовательского ввода с помощью JavaScript и регулярных выражений
- Обработка нескольких значений с использованием коллекций JavaScript
- Управление датами с помощью JavaScript Date API
- Практика: Создание приложения “meal-divider”
- Практика: Расчет общего количества в зависимости от возраста

Регулярные выражения



```
var regexp = /Java/;  
var regexp = /Java/gmi;  
var regexp = new RegExp('Java','gmi');
```

```
'I like JavaScript'.search(/Java/); // 7
```

```
'I like JavaScript'.search(/java/); // -1  
'I like JavaScript'.search(/java/i); // 7
```

```
'ОЙ-Ой-ой'.match(/ой/i);  
// ['ОЙ', index: 0, input: 'ОЙ-Ой-ой']
```

```
'ОЙ-Ой-ой'.match(/ой/ig);  
// ['ОЙ', 'Ой', 'ой']
```

```
'I like JavaScript'.match(/java(script)/i);  
// ["JavaScript", "Script"]
```

```
'I like JavaScript'.match(/love/gi); //null
```

Регулярные выражения



```
'12-34-56'.replace(/-/,':');    // '12:34-56'  
'12-34-56'.replace(/-/g, ':'); // '12:34:56'
```

```
"Василий Пупкин".  
  replace(/(Василий) (Пупкин)/, '$2, $1');  
// Пупкин, Василий
```

```
var i = 0;  
  
"ОЙ-Ой-ой".replace(  
  /ой/gi,  
  function(){return ++i;}  
);  
  
// "1-2-3"
```

```
var str="Читайте про JavaScript на javascript.ru";
var regexp=/javascript/ig;

while (result=regexp.exec(str)){
    console.log(result);
}

// ["JavaScript", index: 12, input: "Читайте про
JavaScript на javascript.ru"]

//["javascript", index: 26, input: "Читайте про
JavaScript на javascript.ru"]
```

```
/@/ig.test("me@mydomain.com");           //true
```

Регулярные выражения



<code>^</code>	начало текста	<code>?</code>	0 или 1
<code>\$</code>	конец текста	<code>*</code>	от 1 до ∞
<code>\b</code>	граница слова	<code>+</code>	от 1 до ∞
<code>\s</code>	пробельный символ	<code>{1-5}</code>	от 1 до 5
<code>\S</code>	не <code>\s</code>	<code>{3}</code>	ровно 3
<code>\w</code>	буква или <code>_</code>	<code>{2,}</code>	от 2 до ∞
<code>\W</code>	не <code>\w</code>	<code> </code>	или (альтернатива)
<code>\d</code>	цифра	<code>()</code>	подшаблон
<code>\D</code>	не цифра	<code>\</code>	экранирование
<code>.</code>	любой символ	<code>/m</code>	многострочный
<code>[abc]</code>	набор символов		
<code>[a-z]</code>	диапазон		
<code>[^abc]</code>	все, кроме abc		

Цвет

```
/#[0-9a-f]{6}/gi.test("#0dff97");
```

Форма

```
/<form(\s|>).*</form>/i
```

Число с дробной частью

```
/^\d+\.\d+$/ig
```


Map, Set, WeakMap, WeakSet (ES6)



```
let s = new Set()
s.add("hello").add("goodbye").add("hello")
s.size === 2
s.has("hello") === true
for (let key of s.values()) console.log(key)
```

```
let m = new Map()
m.set("key1", 42)
m.set("key2", 34)
m.get("key2") === 34
m.size === 2
for (let [ key, val ] of m.entries())
  console.log(key + " = " + val)
```

WeakMap, WeakSet – не хранят указатели

Date



`new Date()`

`new Date(milliseconds)`

`new Date(dateString)`

`new Date(year, month, day, hours, minutes, seconds, milliseconds)`

```
new Date ()
```

```
new Date (86400000)
```

```
new Date ("2017-06-15") //ISO
```

```
new Date ("2017-06") //ISO
```

```
new Date ("2017") //ISO
```

```
new Date ("2017-06-15T11:35:00Z") //ISO UTC
```

```
new Date ("2017-06-15T11:35:00+03:00") //ISO +3
```

```
new Date (2017, 5, 15, 11, 33, 30, 0)
```

```
new Date (2017, 5, 15)
```

```
<p id="demo"></p>

<script>
  d = new Date();
  document.getElementById("demo").innerHTML = d;
  // toString()
</script>
```

toString ()

toUTCString ()

toDateString ()

Date



getDate()	getUTCDate()	setDate()
getDay()	getUTCDay()	
getFullYear()	getUTCFullYear()	setFullYear()
getHours()	getUTCHours()	setHours()
getMilliseconds()	getUTCMilliseconds()	setMilliseconds()
getMinutes()	getUTCMinutes()	setMinutes()
getMonth()	getUTCMonth()	setMonth()
getSeconds()	getUTCSeconds()	setSeconds()
getTime()		setTime()

```
new Date().setFullYear(2015) < new Date() //true
```