

DOM Events

---

JS  
COURSE  
ORT DNIPRO

---

ORTDNIPRO.ORG/JS

# 1. Событийная модель



## Событийно-ориентированная система управления

Каждая из этих вещей делает что-то в ответ на действия пользователя. Можно сказать каждое действие пользователя это **событие**, и на него нужно как-то **отреагировать**.

# События / Events

*Вариантов событий много, задача программиста выбрать нужное*

## HTML DOM Events

**DOM:** Indicates in which DOM Level the property was introduced.

### Mouse Events

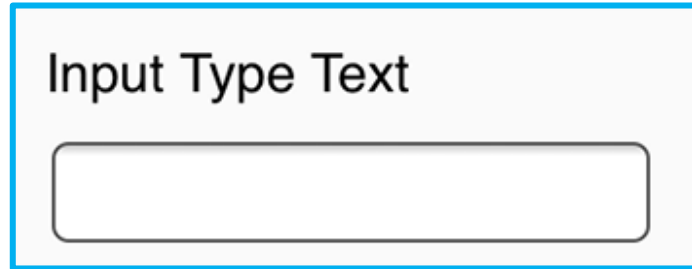
Event	Description	DOM
<a href="#">onclick</a>	The event occurs when the user clicks on an element	2
<a href="#">oncontextmenu</a>	The event occurs when the user right-clicks on an element to open a context menu	3
<a href="#">ondblclick</a>	The event occurs when the user double-clicks on an element	2
<a href="#">onmousedown</a>	The event occurs when the user presses a mouse button over an element	2
<a href="#">onmouseenter</a>	The event occurs when the pointer is moved onto an element	2
<a href="#">onmouseleave</a>	The event occurs when the pointer is moved out of an element	2
<a href="#">onmousemove</a>	The event occurs when the pointer is moving while it is over an element	2
<a href="#">onmouseover</a>	The event occurs when the pointer is moved onto an element, or onto one of its children	2
<a href="#">onmouseout</a>	The event occurs when a user moves the mouse pointer out of an element, or out of one of its children	2
<a href="#">onmouseup</a>	The event occurs when a user releases a mouse button over an element	2

### Keyboard Events

Event	Description	DOM
<a href="#">onkeydown</a>	The event occurs when the user is pressing a key	2
<a href="#">onkeypress</a>	The event occurs when the user presses a key	2
<a href="#">onkeyup</a>	The event occurs when the user releases a key	2

Подробнее: [http://www.w3schools.com/jsref/dom\\_obj\\_event.asp](http://www.w3schools.com/jsref/dom_obj_event.asp)

# События возможные для одних элементов, могут не существовать для других



Поддерживает **ввод с клавиатуры**, события «**фокус**» и «**потеря фокуса**».



Не поддерживает **ввод с клавиатуры**, и событий «**фокус**» и «**потеря фокуса**» для него тоже быть не может.

Однако есть набор событий который поддерживают все элементы: **клик, наведение курсора мыши** и т.д.

## 2. Подписка на события

# Как указать браузеру какую функцию и когда вызывать?

```
2
3  <h1 onclick="eventListener()">Some Content</h1>
4
5  <script>
6
7
8      function eventListener(){
9          console.log('Click detected!');
10     }
11
12 </script>
13
```

*Через соответствующие атрибуты тегов*

# Как указать браузеру какую функцию и когда вызывать?

```
2
3   <h1>Some Content</h1>
4
5   <script>
6
7       let h1Tag = document.querySelector('h1');
8
9       h1Tag.onclick = function(){
10         console.log('Click detected!');
11       }
12
13   </script>
14
```

*Через свойства объектов входящих в дерево документа*

Подробнее: <https://learn.javascript.ru/introduction-browser-events>



# Как указать браузеру какую функцию и когда вызывать?

```
2
3 <h1>Some Content</h1>
4
5 <script>
6
7     let h1Tag = document.querySelector('h1');
8
9     function eventListener_1(){
10         console.log("I'm eventListener_1");
11     }
12
13     function eventListener_2(){
14         console.log("I'm eventListener_2");
15     }
16
17     h1Tag.addEventListener('click', eventListener_1);
18     h1Tag.addEventListener('click', eventListener_2);
19
20     //h1Tag.removeEventListener('click', eventListener_1);
21     //h1Tag.removeEventListener('click', eventListener_2);
22
23 </script>
24
```

При помощи метода **.addEventListener()** можно на одно событие повесить множество обработчиков. А при необходимости и снять обработчик при помощи **.removeEventListener()**.

# Вспоминаем **this**

```
2
3 <h1>Some Content</h1>
4 <p>Some P tag</p>
5
6 ✓ <script>
7
8     let h1Tag = document.querySelector('h1');
9     let pTag = document.querySelector('p');
10
11 ✓     function eventListener(){
12         |     console.log('this in eventListener:', this);
13         |     }
14
15     h1Tag.addEventListener('click', eventListener);
16     pTag.addEventListener('click', eventListener);
17
18
19 </script>
20
```

Функция обработчик становится частью объекта-элемента, и вызывается как его метод. Поэтому ключевое слово **this** в обработчике ссылается на объект который вызвал обработчик события.

# 3. События

onLoad,

onDOMContentLoaded

# Событие window.onload

```
3
4  ✓ window.addEventListener('load', function(e){
5      |     console.log("Event Window.onLoad", e);
6      | });
7
8  ✓ document.addEventListener('DOMContentLoaded', function(e){
9      |     console.log("Event Document.DOMContentLoaded", e);
10     | });
11
```

Событие **onload** (объекта **window**, он же **globalThis**) срабатывает тогда когда загружен (и обработан) HTML документ и все подключаемые файлы, в т.ч изображения, стили т.д.

# Событие document.DOMContentLoaded

```
3
4  ✓ window.addEventListener('load', function(e){
5      |     console.log("Event Window.onLoad", e);
6      | });
7
8  ✓ document.addEventListener('DOMContentLoaded', function(e){
9      |     console.log("Event Document.DOMContentLoaded", e);
10     | });
11
```

Событие **DOMContentLoaded** доступно для объекта **document** через **.addEventListener()** и срабатывает тогда когда загружен HTML документ и JS файлы (завершилась ли загрузка изображений и css-файлов неважно).

# JavaScript в HTML

Разрешить это неудобство (с выполнением кода сразу, а не когда страница полностью загрузится) можно разными способами, например:

1. Разместить весь код в конце документа;
2. Разместить весь код во внешнем файле и подключить его с атрибутом **defer**;
3. Использовать события **onLoad** или **onDOMContentLoaded**.

```
<script defer src="scripts/async.js"></script>
```

*Атрибут **defer** откладывает выполнение скрипта до тех пор, пока вся страница не будет загружена полностью. Работает только для внешних (подключаемых) файлов.*

# 3. Информация о событии

# Информация о событии

*Чтобы обработать событие, недостаточно знать о том, что это – «клик» или «нажатие клавиши». Могут понадобиться детали: координаты курсора, введённый символ и другие, в зависимости от события.*

*Браузер может дать много полезной информации о событии, для этого он создаёт объект, в свойства которого записывает детали произошедшего события. И передаёт этот объект функции обработчику события.*



# Информация о событии

```
2
3  <h1>Some Content</h1>
4
5  <script>
6
7      let h1Tag = document.querySelector('h1');
8
9      function eventListener(e){
10         console.log('Event info:', e);
11     }
12
13     h1Tag.addEventListener('click', eventListener);
14
15 </script>
16
```

*Браузер записывает информацию о событии в объект т.н. «объект события», который передаётся первым аргументом в функцию обработчик события. Если она принимает параметры, т.к. это является необязательным.*

# Информация о событии

*Разные события – разные объекты с информацией о них.*

*В зависимости от типа события, объект с детальной информацией о событии содержит разные наборы полей, например: для событий мыши он содержит координаты курсора, а события клавиатуры он содержит данные о нажатых клавишах.*

Подробнее: <https://learn.javascript.ru/mouse-events-basics>

Подробнее: <https://learn.javascript.ru/keyboard-events>

```

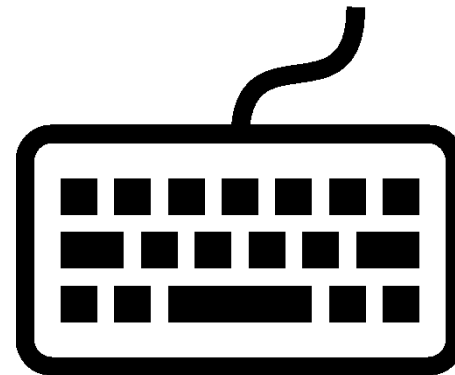
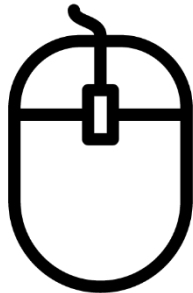
▼ MouseEvent ⓘ
  altKey: false
  bubbles: true
  button: 0
  buttons: 0
  cancelBubble: false
  cancelable: true
  clientX: 83
  clientY: 17
  ctrlKey: false
  currentTarget: null
  defaultPrevented: false
  detail: 1
  eventPhase: 0
  fromElement: null
  isTrusted: true
  isTrusted: true
  layerX: 83
  layerY: 17
  metaKey: false
  movementX: 0
  movementY: 0
  offsetX: 75
  offsetY: 9
  pageX: 83
  pageY: 17
  ▶ path: Array[5]
    relatedTarget: null
    returnValue: true
    screenX: 2003
    screenY: 102
    shiftKey: false
  ▶ sourceCapabilities: InputDeviceCapabilities
  ▶ srcElement: p
  ▶ target: p
    timeStamp: 1314.7900000000002
  ▶ toElement: p
    type: "click"
  ▶ view: Window
    which: 1
    x: 83
    y: 17
  ▶ __proto__: UIEvent

```

# Информация о событии

*Разные события – разные объекты с информацией о них.*

*<= MouseEvent*



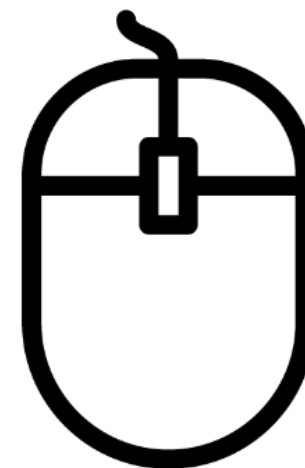
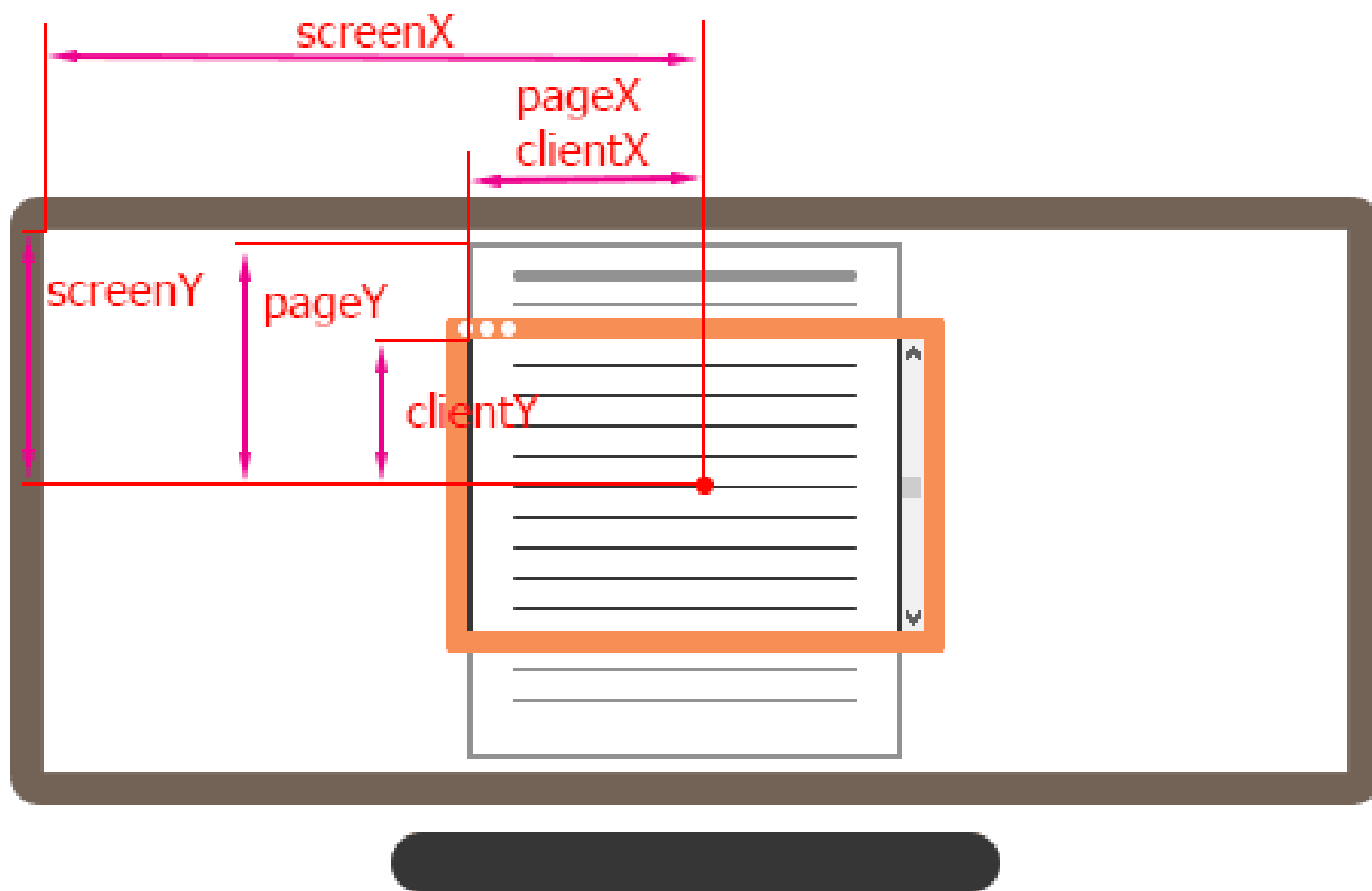
*KeyboardEvent =>*

```

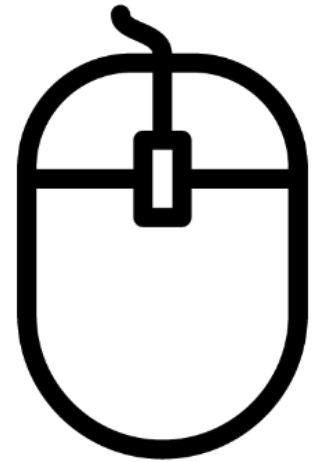
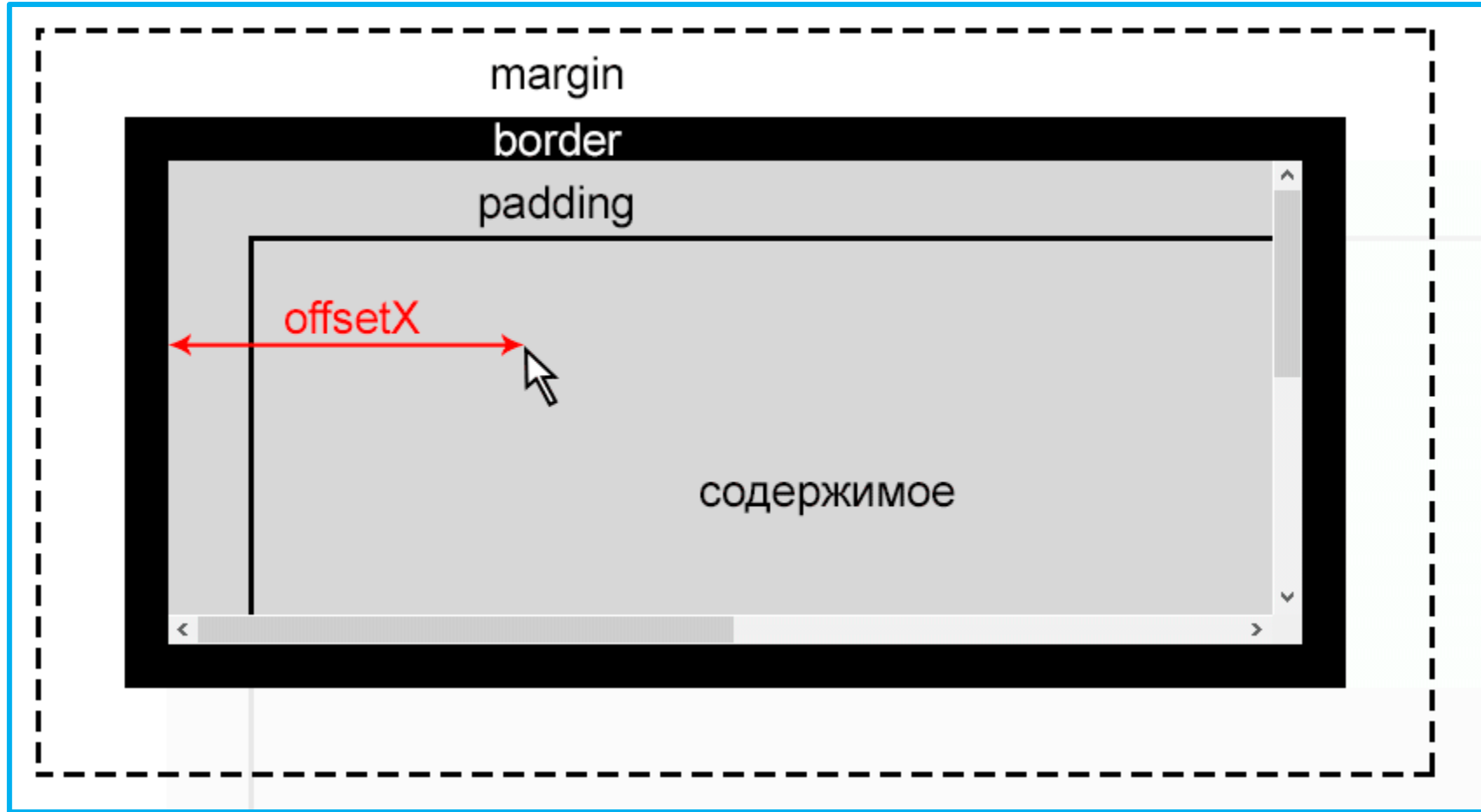
▼ KeyboardEvent ⓘ
  altKey: false
  bubbles: true
  cancelBubble: false
  cancelable: true
  charCode: 97
  code: "KeyA"
  ctrlKey: false
  currentTarget: null
  defaultPrevented: false
  detail: 0
  eventPhase: 0
  isTrusted: true
  isTrusted: true
  keyCode: 97
  keyIdentifier: "U+0041"
  keyLocation: 0
  location: 0
  metaKey: false
  ▶ path: Array[5]
    repeat: false
    returnValue: true
    shiftKey: false
  ▶ sourceCapabilities: InputDeviceCapabilities
  ▶ srcElement: input
  ▶ target: input
    timeStamp: 2079.225
    type: "keypress"
  ▶ view: Window
    which: 97
  ▶ __proto__: UIEvent

```

# Позиция курсора мыши в объекте MouseEvent

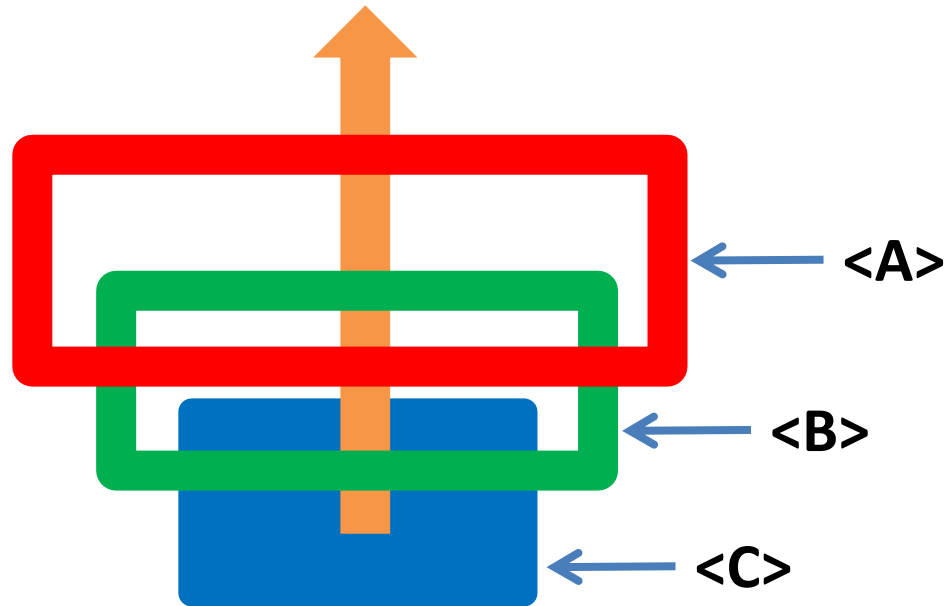


# Позиция курсора мыши в объекте MouseEvent



## 4. всплытие событий

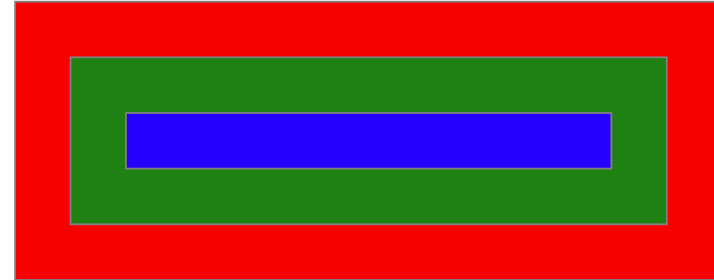
## «Всплытие» событий



При наступлении события обработчики сработав на целевом элементе начинают подниматься **«всплывать»** к предкам элемента.

# «Всплытие» событий

Воспользуйтесь заготовкой:  
[./src/event-bubling-example](#)



```
17
18   function eventListener(e){
19       |   console.log(this.id);
20       |
21       |
22   document.addEventListener('DOMContentLoaded', () => {
23       |
24       |   document.querySelectorAll('div').forEach(item => {
25       |       |   item.addEventListener('click', eventListener);
26       |       |
27       |       |
28       |   });
29   });
```

*Что мы увидим в консоли после клика по синему блоку?*



# e.target

Свойство **.target** (объекта события) содержит ссылку на объект инициатор события, т.е. например тот элемент по которому произошел клик.

```
17
18     function eventListener(e){
19         |     console.log(this.id, e.target.id);
20     }
21
22     document.addEventListener('DOMContentLoaded', () => {
23         |
24         document.querySelectorAll('div').forEach(item => {
25             |     item.addEventListener('click', eventListener);
26         });
27     });
28
29
```

Подробнее: <https://learn.javascript.ru/bubbling-and-capturing>

# Всплытие можно остановить

```
17
18 function eventListener(e){
19     console.log(this.id, e.target.id);
20     e.stopPropagation();
21 }
22
23 document.addEventListener('DOMContentLoaded', () => {
24
25
26     document.querySelectorAll('div').forEach(item => {
27         item.addEventListener('click', eventListener, false);
28     });
29
30 });
31
```

***`e.stopPropagation()`** – останавливает всплытие событий.*

Подробнее: <https://learn.javascript.ru/bubbling-and-capturing>

# 5. Действие по умолчанию

# Действия по умолчанию

*У некоторых элементов есть встроенная реакция на событие, или по другому действие по умолчанию.*

**Например:**

- 1. Для ссылок действие по умолчанию переход на другую страницу;*
  - 2. Для кнопок внутри формы действие по умолчанию – отправить форму на сервер;*
  - 3. Двойной клик по тексту – выделяет его фрагмент.*
- и т.д.*

# Отмена действия по умолчанию

```
2
3  <a href="https://itc.ua">Site ITC.ua</a>
4
5  <script>
6
7      let aTag = document.querySelector('a');
8
9      aTag.addEventListener('click', function(e){
10         e.preventDefault();
11     });
12
13 </script>
14
```

***e.preventDefault()*** – (метод объекта с информацией о событии) отменяет действие по умолчанию (если такое предусмотрено).

# Не путайте!

**e.preventDefault()** – отменяет действие по умолчанию (как то переход по ссылке, отправка формы и т.д.).

**e.stopPropagation()** – останавливает всплытие события, т.е. после вызова этого метода элементы далее по иерархии уже не получают уведомление о событии.

## 6. Немного практики

# Рисование, Графика, Canvas

```
31 <canvas id="paint-canvas"></canvas>
32 <script>
33
34     var canvas      = document.getElementById("paint-canvas");
35     canvas.width     = canvas.clientWidth;
36     canvas.height    = canvas.clientHeight;
37
38     var context      = canvas.getContext("2d");
39
40     context.moveTo(200, 200);
41     context.lineTo(300, 250);
42     context.lineTo(200, 300);
43     context.closePath();
44     context.stroke();
45
46 </script>
```

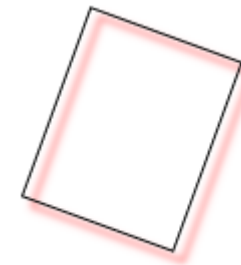
Тег **canvas** – представляет собой «холст», прямоугольную область в которой можно рисовать. **Контекст canvas'a** – объект который содержит множество методов для рисования на «холсте».



# Рисование, Графика, Canvas

Рисование на **canvas**'е основано на отрисовке примитивов.

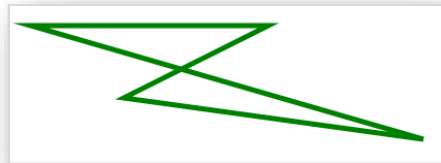
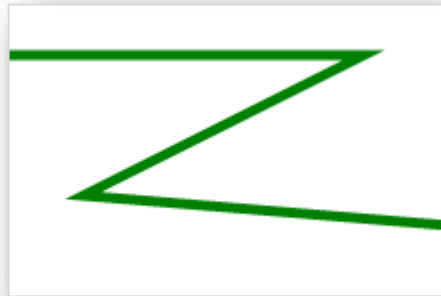
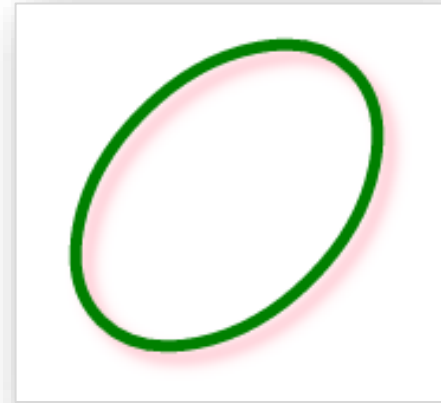
- 1) Штриховых (контурных фигур) – в названии методов и свойств есть слово **stroke**;
- 2) Заполненных фигур, в названии методов и свойств есть слово **fill**;
- 3) Наложении спецэффектов (тени, развороты, искажения и т.п.).



# Рисование, Графика, Canvas

Примитивы можно рисовать при помощи функций-заготовок: прямоугольник (**rect()**), эллипс (**ellipse()**) и т.п.

Либо самостоятельно задав контур фигуры состоящей из множества линий. Для этого есть функции **beginPath()** и **closePath()** – для случаев когда нужно замкнуть контур (между первой и последней точкой фигуры).



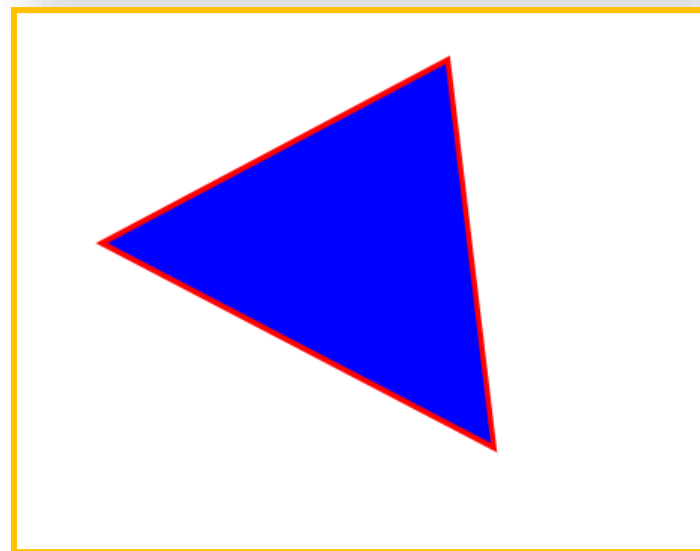
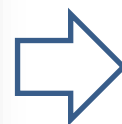
# Рисование примитивов

```
context.beginPath();  
context.moveTo(175, 225);  
context.lineTo(400, 113);  
context.lineTo(430, 350);  
//context.closePath();  
context.stroke();
```

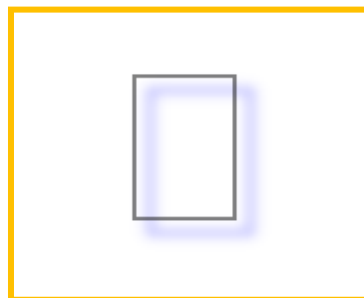


```
context.beginPath();  
context.moveTo(175, 225);  
context.lineTo(400, 113);  
context.lineTo(430, 350);  
//context.closePath();  
context.fill();
```

```
context.beginPath();  
context.moveTo(175, 225);  
context.lineTo(400, 113);  
context.lineTo(430, 350);  
context.closePath();  
context.lineWidth = 7;  
context.strokeStyle = "red";  
context.fillStyle = "blue";  
context.stroke();  
context.fill();
```



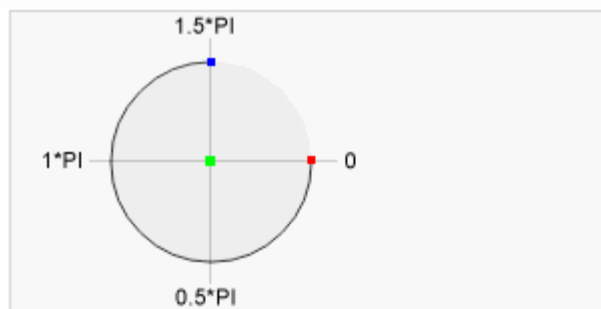
```
context.rect(300, 200, 50, 80);  
context.shadowBlur = 10;  
context.shadowOffsetY = 8;  
context.shadowOffsetX = 8;  
context.shadowColor = "blue";  
context.stroke();
```



# Рисование примитивов

```
context.strokeStyle = "green";  
context.rect(300, 200, 50, 80);  
//rect(x,y, width, height)  
context.stroke();
```

```
context.arc(200,300,45, 1.8*Math.PI, 2*Math.PI, false);  
//arc(x, y, radius, startAngle, finishAngle, direction);  
context.stroke();
```



Center	arc( <b>100</b> ,75,50,0*Math.PI,1.5*Math.PI)
Start angle	arc(100,75,50, <b>0</b> ,1.5*Math.PI)
End angle	arc(100,75,50,0*Math.PI, <b>1.5*Math.PI</b> )

*Прямоугольник и дуга*

# Свойства (графические атрибуты «холста»)

## Paths

Method	Description
<a href="#">fill()</a>	Fills the current drawing (path)
<a href="#">stroke()</a>	Actually draws the path you have defined
<a href="#">beginPath()</a>	Begins a path, or resets the current path
<a href="#">moveTo()</a>	Moves the path to the specified point in the canvas, without creating a line
<a href="#">closePath()</a>	Creates a path from the current point back to the starting point
<a href="#">lineTo()</a>	Adds a new point and creates a line to that point from the last specified point in the canvas
<a href="#">clip()</a>	Clips a region of any shape and size from the original canvas
<a href="#">quadraticCurveTo()</a>	Creates a quadratic Bézier curve
<a href="#">bezierCurveTo()</a>	Creates a cubic Bézier curve
<a href="#">arc()</a>	Creates an arc/curve (used to create circles, or parts of circles)
<a href="#">arcTo()</a>	Creates an arc/curve between two tangents
<a href="#">isPointInPath()</a>	Returns true if the specified point is in the current path, otherwise false

## Transformations

Method	Description
<a href="#">scale()</a>	Scales the current drawing bigger or smaller
<a href="#">rotate()</a>	Rotates the current drawing
<a href="#">translate()</a>	Remaps the (0,0) position on the canvas
<a href="#">transform()</a>	Replaces the current transformation matrix for the drawing

Подробнее: [http://www.w3schools.com/tags/ref\\_canvas.asp](http://www.w3schools.com/tags/ref_canvas.asp)

# «Paint» на JavaScript



## «Paint» на JavaScript

Простой аналог программы «**Paint**» на базе **JavaScript** и **canvas**.

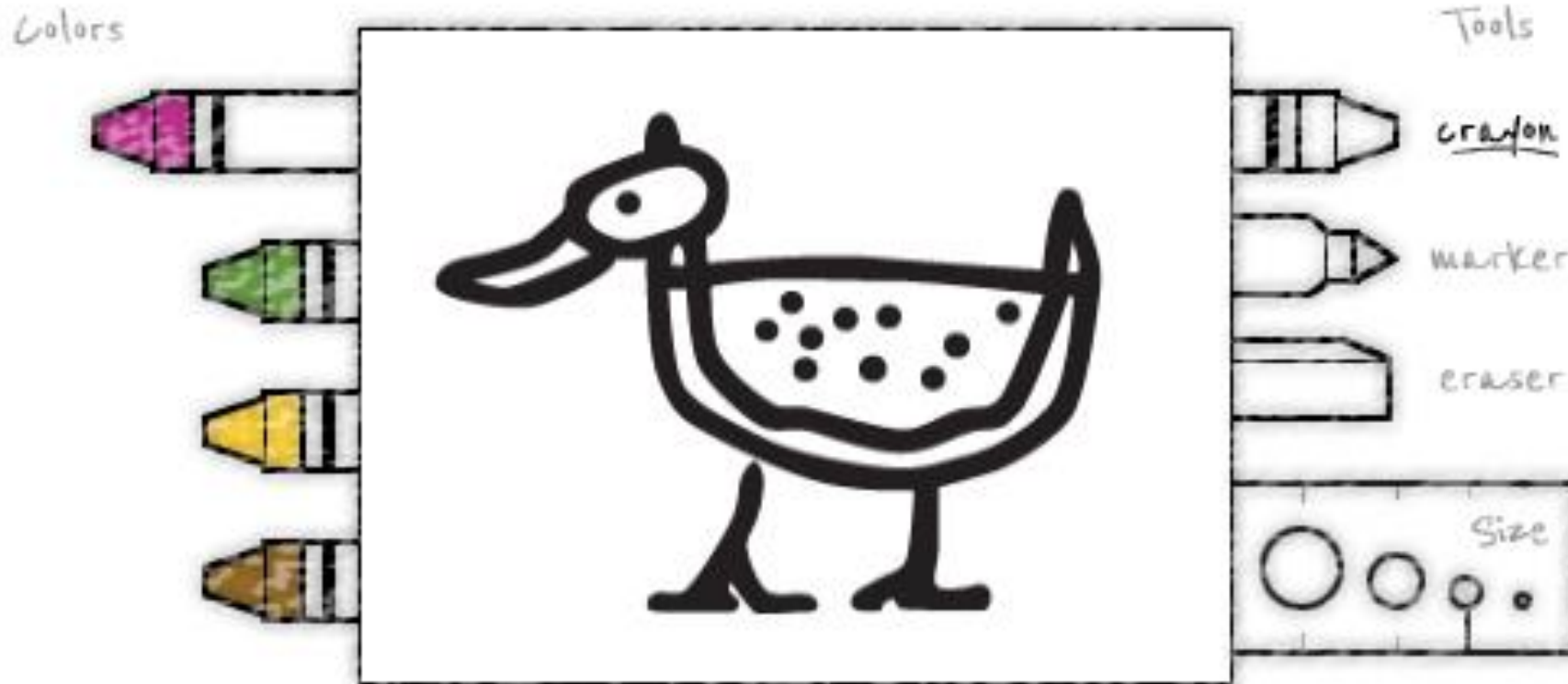
Воспользуйтесь шаблоном в репозитории занятия:

[./src/paint-example](#)

**Будет полезным**



# JavaScript + Canvas = Paint



<http://www.williammalone.com/articles/create-html5-canvas-javascript-drawing-app/>

**К следующему занятию будет  
полезно почитать о...**













**К следующему занятию...**

**Формы, элементы ввода,  
регулярные выражения и  
валидация данных**

**Домашнее задание**  
**/сделать**

# Домашнее задание #Н.1

☐ Сортировать по возрастанию цены ☐ Сортировать по уменьшению цены

 <p><b>Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops</b></p> <p>Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in t...</p> <p><b>\$109.95</b></p>	 <p><b>Mens Casual Premium Slim Fit T-Shirts</b></p> <p>Slim-fitting style, contrast raglan long sleeve, three-button henley placket, light weight &amp; soft fa...</p> <p><b>\$22.3</b></p>	 <p><b>Mens Cotton Jacket</b></p> <p>great outerwear jackets for Spring/Autumn/Winter, suitable for many occasions, such as working, hiki...</p> <p><b>\$55.99</b></p>	 <p><b>Mens Casual Slim Fit</b></p> <p>The color could be slightly different between on the screen and in practice. / Please note that body...</p> <p><b>\$15.99</b></p>
 <p><b>John Hardy Women's Legends Naga Gold &amp; Silver Dragon Station Chain Bracelet</b></p> <p>From our Legends Collection, the Naga was inspired by the mythical water dragon that protects the oc...</p> <p><b>\$695</b></p>	 <p><b>Solid Gold Petite Micropave</b></p> <p>Satisfaction Guaranteed. Return or exchange any order within 30 days.Designed and sold by Hafeez Cen...</p> <p><b>\$168</b></p>	 <p><b>White Gold Plated Princess</b></p> <p>Classic Created Wedding Engagement Solitaire Diamond Promise Ring for Her. Gifts to spoil your love ...</p> <p><b>\$9.99</b></p>	 <p><b>Pierced Owl Rose Gold Plated Stainless Steel Double</b></p> <p>Rose Gold Plated Double Flared Tunnel Plug Earrings. Made of 316L Stainless Steel...</p> <p><b>\$10.99</b></p>
 <p><b>WD 2TB Elements Portable External Hard Drive - USB 3.0</b></p>	 <p><b>SanDisk SSD PLUS 1TB Internal SSD - SATA III 6 Gb/s</b></p>	 <p><b>Silicon Power 256GB SSD 3D NAND A55 SLC Cache Performance Boost SATA III 2.5</b></p>	 <p><b>WD 4TB Gaming Drive Works with Playstation 4 Portable External Hard Drive</b></p>

Воспользуйтесь сервисом:

<https://fakestoreapi.com/>

А именно разделом <https://fakestoreapi.com/products> для формирования страницы со списком товаров.