# События / Events


JavaScript
Courses

vk.com/js.courses

js.courses.dp.ua/files

# Что общего у этих вещей?

# Система управления основанная на событиях

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

# События / Events

*В программировании обработка событий основана на функциях. Поскольку функции хорошо подходят для того чтобы многократно (неизвестно заранее сколько) выполнять один и тот же фрагмент кода.*

# События / Events

**Смысл событий в JS** - сказать браузеру: «когда произойдёт клик по элементу, то выполни вот эту функцию»;

```
1  <html>
2      <head>
3          <script>
4              function on_event(){
5                  alert("Функция вызвана!");
6              }
7          </script>
8      </head>
9      <body>
10         <p onclick="on_event();">Text Text Text</p>
11     </body>
12 </html>
```

# События / Events

**Событийная модель** – *подход в программировании, когда действия программы определяются событиями, как правило действиями пользователя (мышь, клавиатура, сенсор), сообщениями от других программ и/или операционной системы.*

***Событие*** *– действие о котором браузер должен уведомить нашу программу;*

***Подписаться на событие*** *– указать браузеру, что «при клике нужно вызвать функцию ABC()»;*

***Обработчик события*** *– функция которая будет вызываться при наступлении события;*

***Слушать событие*** *– тоже самое, что и ждать наступления события.*

# События / Events

*Логично, что программа не может отреагировать на абсолютно все возможные события, который могут произойти.*

*Программа может отреагирует (и то, если об этом позаботился программист) только на те события которые предусмотрены операционной системой (браузером).*

*Если в операционной системе нет поддержки датчика температуры, то программа не узнает, что пользователь подул на компьютер феном.* ☺️

*Вариантов событий много, задача программиста выбрать нужное.*

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

### Keyboard Events

| Event | Description | DOM |
|---|---|---|
| onkeydown | The event occurs when the user is pressing a key | 2 |
| onkeypress | The event occurs when the user presses a key | 2 |
| onkeyup | The event occurs when the user releases a key | 2 |

http://www.w3schools.com/jsref/dom_obj_event.asp

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

```html
<html>
    <head>
        <script>
            function on_focus(){ console.log("focus"); };
            function on_blur(){ console.log("blur"); };
            function on_key(){ console.log("key"); }
        </script>
    </head>
    <body>
        <input type="text" onfocus="on_focus();" onblur="on_blur();" onkeypress="on_key();">
    </body>
</html>
```

```html
<html>
    <head>
        <script>
            function on_focus(){ console.log("focus"); };
            function on_blur(){ console.log("blur"); };
            function on_key(){ console.log("key"); }
        </script>
    </head>
    <body>
        <div onfocus="on_focus();" onblur="on_blur();" onkeypress="on_key();">Text Text Text</div>
    </body>
</html>
```

http://www.w3schools.com/jsref/dom_obj_event.asp

# Как сказать браузеру какую функцию и когда вызывать?

***Через атрибуты HTML-элементов:***

```
<html>
    <head>
        <script>
            function on_event(){
                alert("Функция вызвана!");
            }
        </script>
    </head>
    <body>
        <p onclick="on_event();" onmouseover="on_event();">Text Text Text</p>
    </body>
</html>
```

*Недостаток - JavaScript код в HTML-разметке.*

# Как сказать браузеру какую функцию и когда вызывать?

***Через свойства объектов входящих в дерево***

```
<html>
    <head>
        <script>
            var func = function(){
                alert("Зафиксировано событие");
            }

            window.onload = function(){
                var p = document.querySelector("p");
                p.onclick = func;
            }
        </script>
    </head>
    <body>
        <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit</p>
    </body>
</html>
```

*Недостаток - можно подключить максимум один обработчик события.*

# Как сказать браузеру какую функцию и когда вызывать?

***При помощи функции***

```
<html>
    <head>
        <script>
            function handler1(){
                alert("Обработчик №1");
            }

            function handler2(){
                alert("Обработчик №2");
            }

            window.onload = function(){
                var p = document.querySelector("p");
                p.addEventListener("click", handler1);
                p.addEventListener("click", handler2);

                //Подписку можно и отменить, функцией removeEventListener()
                //p.removeEventListener("click", handler1);
                //p.removeEventListener("click", handler1);
            }
        </script>
    </head>
    <body>
        <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit</p>
    </body>
</html>
```

*При помощи **addEventListener()** можно на одно событие повесить множество обработчиков.*

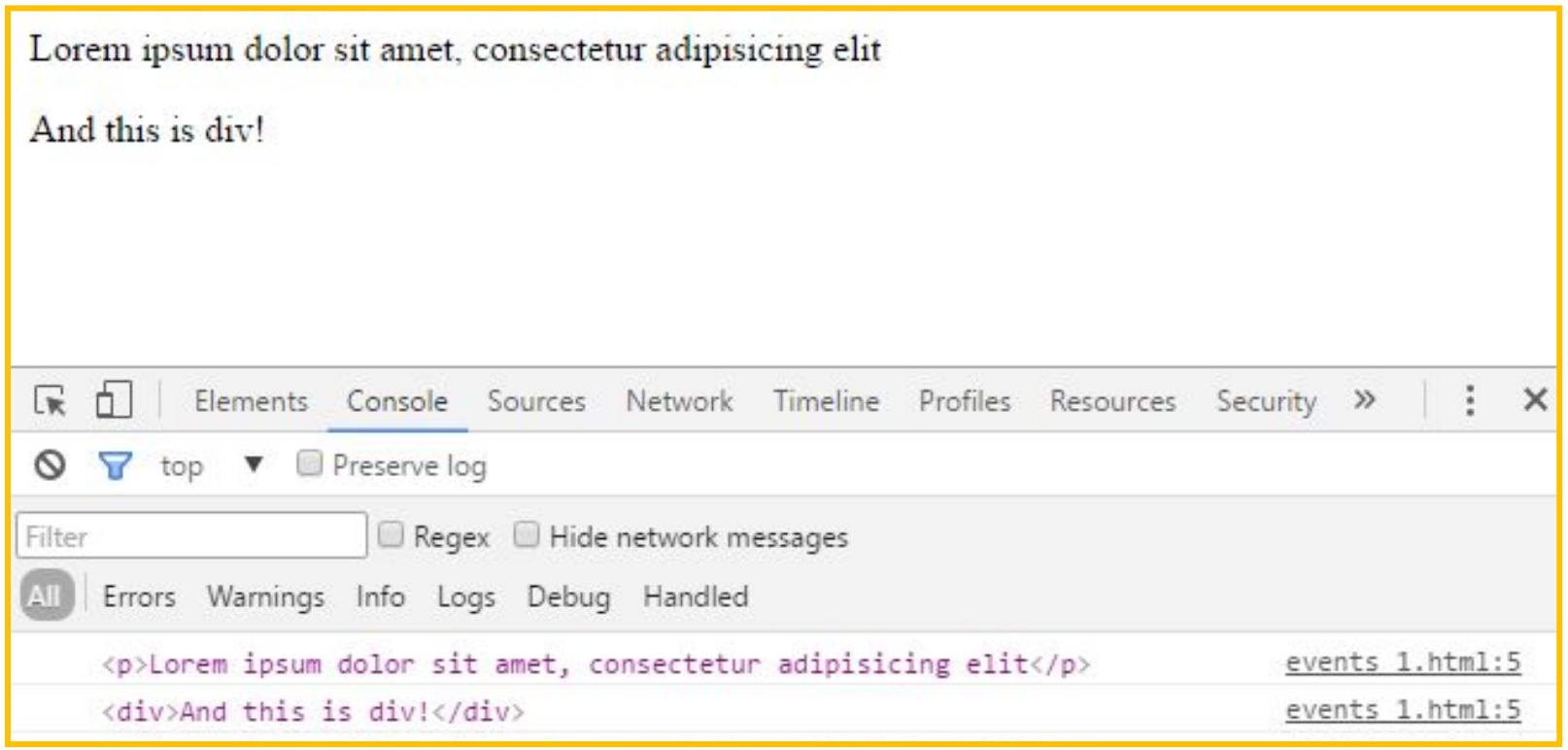
# Вспоминаем this

*Функция обработчик становиться частью объекта-элемента, и вызывается как его метод. Поэтому ключевое слово **this** в обработчике ссылается на объект с которым произошло событие.*

```
1  <html>
2      <head>
3          <script>
4              function handler(){
5                  console.log(this);
6              }
7
8              window.onload = function(){
9                  var p = document.querySelector("p");
10                 p.onclick = handler;
11                 var d = document.querySelector("div");
12                 d.onclick = handler;
13             }
14         </script>
15     </head>
16     <body>
17         <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit</p>
18         <div>And this is div!</div>
19     </body>
20 </html>
```

# События / Events

*Функция обработчик становиться частью объекта-элемента, и вызывается как его метод. Поэтому ключевое слово **this** в обработчике ссылается на объект с которым произошло событие.*



# Вспоминаем

*Ключевое слово* ***this*** *– ссылка на сам объект из функции которая входит в его состав. Для функций явно не входящих в объект* ***this*** *ссылается на объект* ***window****.*

this

```
<script>
    function handler(){
        console.log(this);
    }

    var ob = {
        some_data: 7751
    }

    ob.func = handler;

    handler();
    ob.func();
</script>
```

```
▶ Window {external: Object, chrome: Object, document: document, ob: Object, speechSynthesis: SpeechSynthesis…}
▼ Object {some_data: 7751}
  ▶ func: function handler()
    some_data: 7751
  ▶ __proto__: Object
```

# 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.*

**https://developer.mozilla.org/en-US/docs/Web/API/MouseEvent**

# Информация о событии

```
1  <html>
2      <head>
3          <script>
4              function handler(e){ console.dir(e); }
5
6              window.onload = function(){
7                  document.querySelector("button").onclick = handler;
8              }
9          </script>
10     </head>
11     <body>
12         <button>Click Me!</button>
13     </body>
14 </html>
```

???

*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.*

https://developer.mozilla.org/en-US/docs/Web/API/MouseEvent

https://developer.mozilla.org/en-US/docs/Web/API/KeyboardEvent

# Информация о событии

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

```
1  <html>
2      <head>
3          <script>
4              function handler(e){ console.dir(e); }
5
6              window.onload = function(){
7                  document.querySelector("p").onclick = handler;
8                  document.querySelector("input").onkeypress = handler;
9              }
10         </script>
11     </head>
12     <body>
13         <p>Text Text Text</p>
14         <input type="text">
15     </body>
16 </html>
```

???

https://developer.mozilla.org/en-US/docs/Web/API/MouseEvent

https://developer.mozilla.org/en-US/docs/Web/API/KeyboardEvent

# Информация о событии

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


*onclick*


*onkeypress*

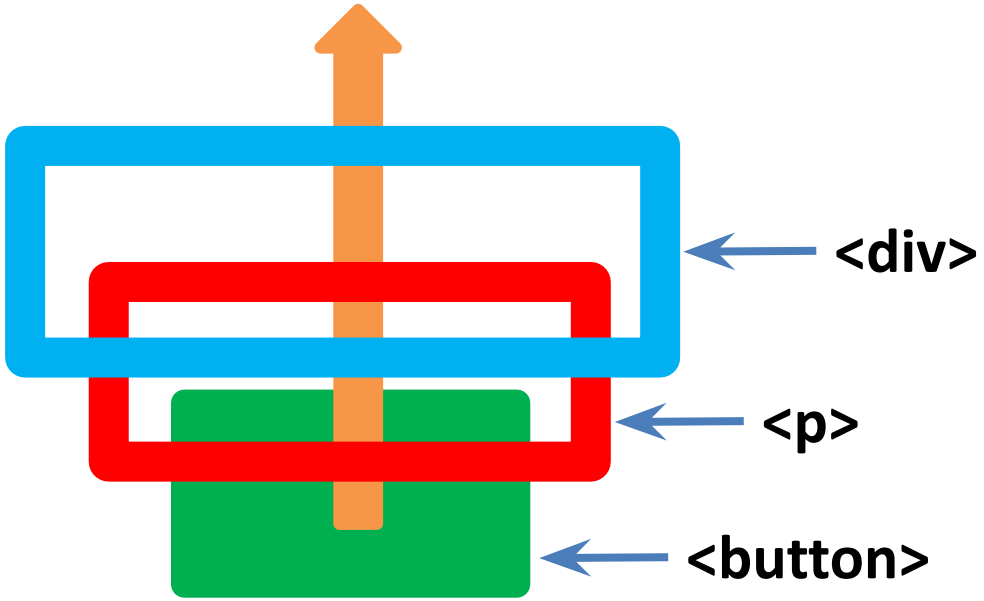
# События / Events

```
<html>
    <head>
        <script>
            function handler(){
                console.log("Event on " + this);
            }

            window.onload = function(){
                var b = document.querySelector("button");
                var p = document.querySelector("p");
                var d = document.querySelector("div");

                b.onclick = handler;
                p.onclick = handler;
                d.onclick = handler;
            }
        </script>
    </head>
    <body>
        <div>
            <p>
                <button>Click Me!</button>
            </p>
        </div>
    </body>
</html>
```

?!?

*Что мы увидим в консоли после клика на кнопку?*

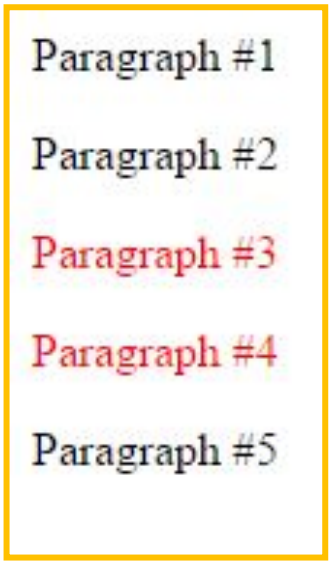
# Всплытие события/ Events bubble




*При наступлении события обработчики сначала срабатывают на самом вложенном элементе (т.е. на том, по которому кликнули), затем на его родителе, затем выше и так далее, вверх по дереву, по цепочке вложенности.*

# Зачем нужно

```html
<html>
    <head>
        <script>
            function handler(e){
                e.target.style.color = "red";
            }

            window.onload = function(){
                document.querySelector("div").onclick = handler;
            }
        </script>
    </head>
    <body>
        <div>
            <p>Paragraph #1</p>
            <p>Paragraph #2</p>
            <p>Paragraph #3</p>
            <p>Paragraph #4</p>
            <p>Paragraph #5</p>
        </div>
    </body>
</html>
```




*Родительский элемент может обрабатывать событие за всех потомков.*

# Зачем нужно всплытие?

**Разные типы элементов – разные события, но**

```
1  <html>
2      <head>
3          <script>
4              function on_key(){ console.log("key"); }
5          </script>
6      </head>
7      <body onkeypress="on_key()">
8          <p>
9          Duis neque mi, rhoncus in urna quis, congue pr
           molestie et dolor eu eleifend. Aliquam sed jus
           Suspendisse non efficitur justo. Donec feugiat
           tincidunt non vitae lorem. Mauris vitae conseq
10         </p>
11     </body>
12 </html>
```

*Когда элементов ввода на странице нет, но нужно получать информацию с клавиатуры.*

# Всплытие можно

```html
<html>
    <head>
        <script>
            function handler(e){
                console.log("Event on " + this);
                e.stopPropagation();
            }

            window.onload = function(){
                var b = document.querySelector("button");
                var p = document.querySelector("p");
                var d = document.querySelector("div");

                b.onclick = handler;
                p.onclick = handler;
                d.onclick = handler;
            }
        </script>
    </head>
    <body>
        <div>
            <p>
                <button>Click Me!</button>
            </p>
        </div>
    </body>
</html>
```

???

```
Event on [object HTMLButtonElement]        event buble stop.html:5
```

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

# Немного практики

```
<html>
    <head>
    <style>
        body { margin: 0; padding: 0; }
    </style>
        <script>
            function handler(e){
                var x = document.createElement("img");
                x.src = "http://js.courses.dp.ua/files/etc/target.svg";
                x.width = "30";
                x.style.position = "absolute";
                x.style.top = e.pageY - 14;
                x.style.left = e.pageX - 14;
                document.body.appendChild(x);
                console.log(e.clientX, e.clientY);
            }

            window.onload = function(){
                document.querySelector("body").onclick = handler;
            }
        </script>
    </head>
    <body>
        <img src="http://js.courses.dp.ua/files/etc/worldmap.jpg" width="100%">
    </body>
</html>
```

???

# Немного практики

# 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 ум

```html
<html>
    <head>
        <script>
            function handler(e){
                e.preventDefault();
            }

            window.onload = function(){
                document.querySelector("a").onclick = handler;
            }
        </script>
    </head>
    <body>
        <a href="http://www.itc.ua">This is a link to ITC.ua</a>
    </body>
</html>
```

```html
<html><head><script>
        function f(e){ e.preventDefault(); }

        window.onload = function(){
            document.querySelector("p").onselectstart = f;
        }
</script></head><body>
        <p>Lorem ipsum dolor sit amet.</p>
</body></html>
```

***.preventDefault()*** *– отменяет действие по умолчанию (если такое предусмотрено).*

# Не путайте!

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

***.stopPropagation()*** *– останавливает всплытие события, т.е. после вызова этой функции элементы-родители уже не получат уведомление о событии.*

# event.target

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

```
<html>
    <head>
        <script>
            function handler(e){
                console.log("Event on " + this + "; Target: " + e.target);
            }

            window.onload = function(){
                var b = document.querySelector("button");
                var p = document.querySelector("p");
                var d = document.querySelector("div");

                b.onclick = handler;
                p.onclick = handler;
                d.onclick = handler;
            }
        </script>
    </head>
    <body>
        <div>
            <p>
                <button>Click Me!</button>
            </p>
        </div>
    </body>
</html>
```

# event.target

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

```
Event on [object HTMLButtonElement]; Target: [object HTMLButtonElement]        event buble param.html:5
Event on [object HTMLParagraphElement]; Target: [object HTMLButtonElement]     event buble param.html:5
Event on [object HTMLDivElement]; Target: [object HTMLButtonElement]           event buble param.html:5
```

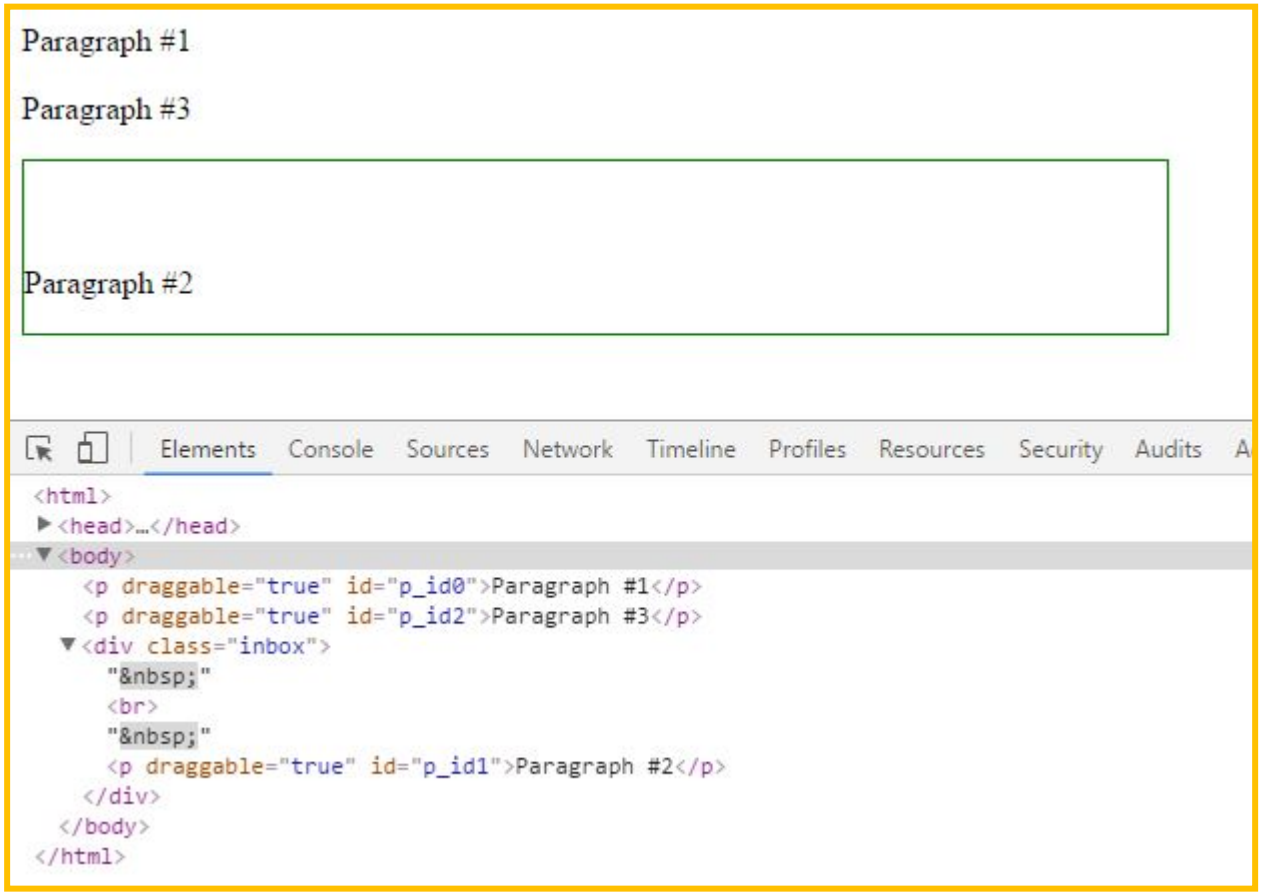
# Drag&Drop

```
<html><head><style>
div.inbox {border: 1px solid green; width: 30%}
</style><script>
function allowDrop(e) { e.preventDefault(); }

function drag(e) { e.dataTransfer.setData("text", e.target.id); }

function drop(e) {
    e.preventDefault();
    var data = e.dataTransfer.getData("text");
    e.target.appendChild(document.getElementById(data));
}

window.onload = function(){
    var ps = document.querySelectorAll("p");
    for(var i = 0; i < ps.length; i++){
        ps[i].ondragstart = drag;
        ps[i].id = "p_id" + i;
    }

    document.querySelector("div").ondrop = drop;
    document.querySelector("div").ondragover = allowDrop;
}

</script></head><body>

<p draggable="true">Paragraph #1</p>
<p draggable="true">Paragraph #2</p>
<p draggable="true">Paragraph #3</p>

<div class="inbox"> <br> </div>

</body></html>
```

*События поддерживающие перетаскивание элементов появились только в HTML5*

http://www.w3schools.com/html/html5_draganddrop.asp

# Drag&Drop




*События поддерживающие перетаскивание элементов появились только в HTML5*

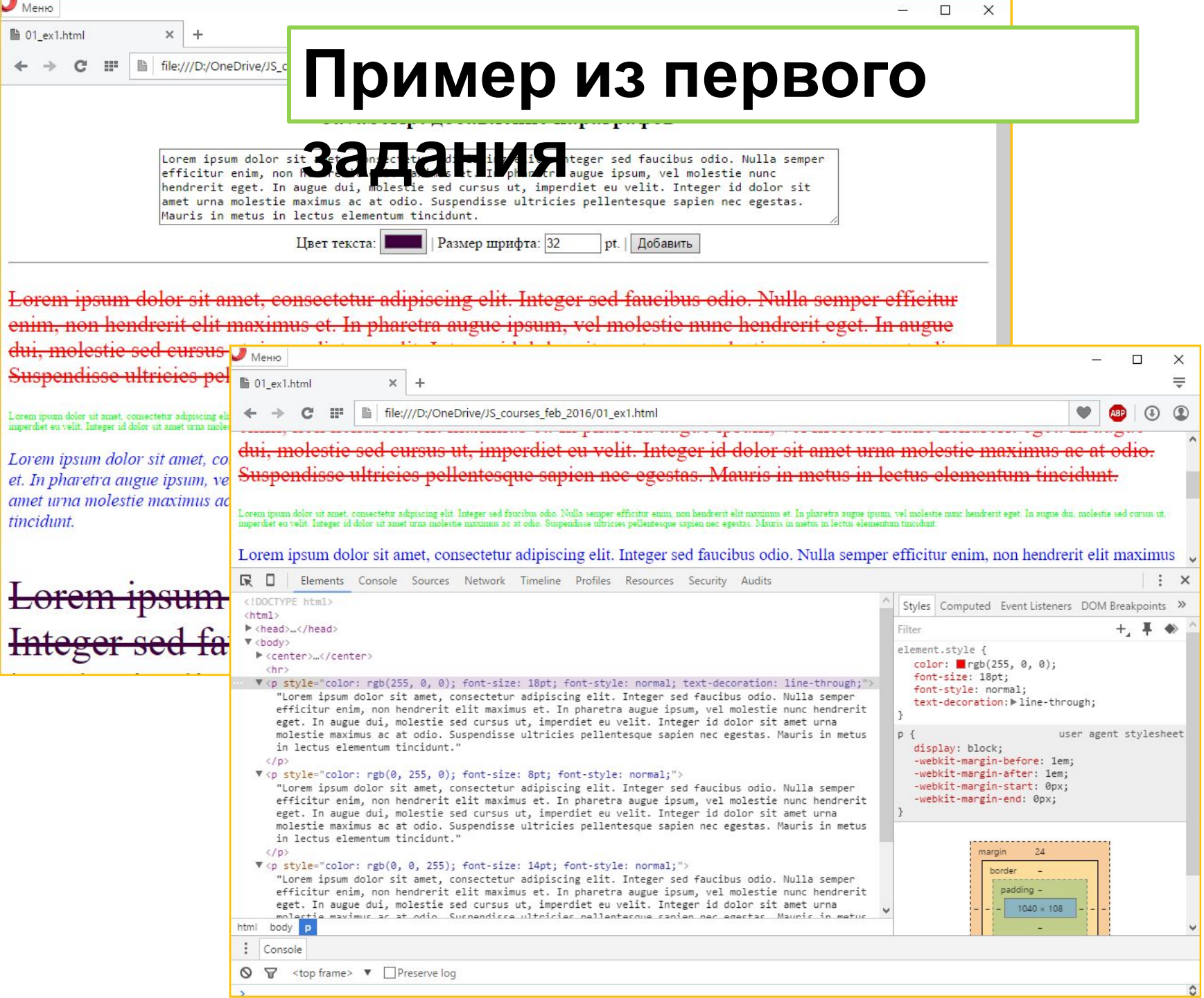
http://www.w3schools.com/html/html5_draganddrop.asp

```
<!DOCTYPE html>
<html>
<head>
<script>
    window.onload = function(){
        document.querySelector("button").onclick = function(){
            var new_element                 = document.createElement("p");
                new_element.innerHTML       = document.querySelector("textarea").value;
                new_element.style.color     = document.querySelector("#color").value;
                new_element.style.fontSize  = document.querySelector("#fontsize").value + "pt";

                new_element.onclick         = function(event_args){
                    event_args.target.style.textDecoration = "line-through";
                };
                new_element.ondblclick      = function(event_args){
                    event_args.target.remove();
                };
                new_element.onmouseover     = function(event_args){
                    event_args.target.style.backgroundColor = '#E0E0E0';
                };
                new_element.onmouseleave    = function(event_args){
                    event_args.target.style.backgroundColor = '#FFFFFF';
                };
            document.body.appendChild(new_element);
        }
    }
</script>
</head>
<body>
    <center>
        <h2>JavaScript Добавление параграфов</h2>
        <textarea cols='100' rows='5'></textarea>
        <div>
            Цвет текста: <input type='color' id='color'>
            | Размер шрифта: <input type='number' id='fontsize' min='8' max='48' value='18'> pt.
            | <button>Добавить</button>
        </div>
    </center>
    <hr>
</body>
</html>
```

# Пример из первого задания

#Первый пример

# Пример из первого задания



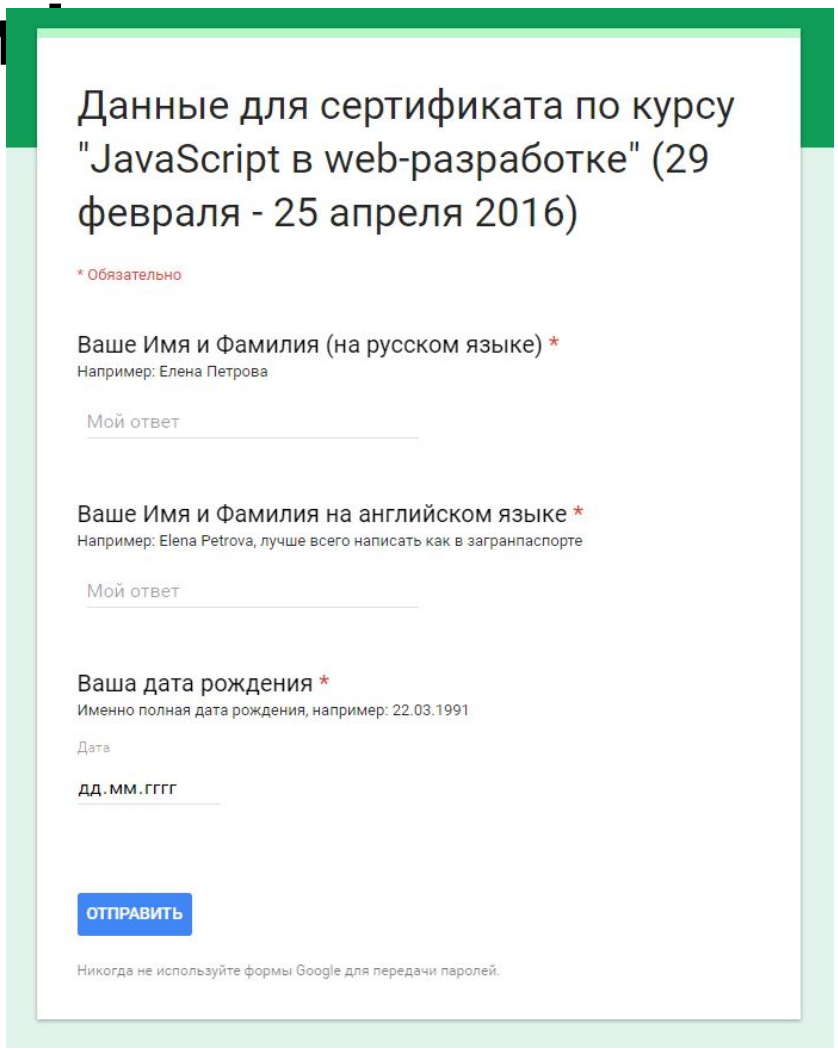

#Первый пример

# Домашнее задание

*1. Изменить программу, так чтобы была возможность ставить только одну метку. Предыдущая должна исчезать после установки новой.*

*2. После каждой установки новой метки выводить alert с расстоянием от метки до Днепропетровска. Выводить в пикселях (это на троечку с минусом), выводить в километрах (это на пять с плюсом).*

# Данные для получения серти



Данные для сертификата по курсу "JavaScript в web-разработке" (29 февраля - 25 апреля 2016)

* Обязательно

Ваше Имя и Фамилия (на русском языке) *
Например: Елена Петрова

Мой ответ

Ваше Имя и Фамилия на английском языке *
Например: Elena Petrova, лучше всего написать как в загранпаспорте

Мой ответ

Ваша дата рождения *
Именно полная дата рождения, например: 22.03.1991

Дата

дд.мм.гггг

ОТПРАВИТЬ

Никогда не используйте формы Google для передачи паролей.

https://docs.google.com/forms/d/18tzSdcm8AR9cjk8O2uguWHnf_3LcAbU94e3fRyE1WJ0/viewform