

# JavaScript & jQuery

# Подключение

```
<script src="/path/to/script.js"></script>
```

```
<script src="file.js">  
  alert(1); // так как указан src, то внутренняя часть тега игнорируется  
</script>
```

```
<script>  
  alert(1);  
</script>
```

# Переменные и типы

```
x = 1;           // число
x = "Тест";     // строка, кавычки могут быть одинарные или двойные
x = true;       // булево значение true/false
x = null;       // спец. значение (само себе тип)
x = undefined; // спец. значение (само себе тип)
```

# Взаимодействие

```
var userName = prompt("Введите имя?", "Василий");
var isTeaWanted = confirm("Вы хотите чай?");

alert("Посетитель: " + userName);
alert("Чай: " + isTeaWanted);
```

# Особенности операторов

```
alert(1 + 2); // 3, число  
alert('1' + 2); // '12', строка  
alert(1 + '2'); // '12', строка
```

```
alert(0 == false); // true  
alert(true > 0); // true
```



# Логические операторы

```
alert(true && true); // true  
alert(false && true); // false  
alert(true && false); // false  
alert(false && false); // false
```

```
alert(0 && 1); // 0  
alert(1 && 2 && 3); // 3  
alert(null || 1 || 2); // 1
```

```
alert(!true); // false  
alert(!0); // true
```

```
alert(!"строка"); // true  
alert(!null); // false
```

# Циклы

```
// 1
□ while (условие) {
  ...
}

// 2
□ do {
  ...
} while (условие);

// 3
□ for (var i = 0; i < 10; i++) {
  ...
}
```

# Switch

```
var age = prompt('Ваш возраст', 18);

□ switch (age) {
  case 18:
    alert( 'Никогда не работает' ); // результат prompt -
                                     // строка, а не число

  case "18": // вот так - работает!
    alert( 'Вам 18 лет!' );
    break;

  default:
    alert( 'Любое значение, не совпавшее с case' );
}
```

# ФУНКЦИИ

```
// function имя(список параметров) { тело }  
function sum(a, b) {  
    var result = a + b;  
  
    return result;  
}  
  
// использование:  
alert( sum(1, 2) ); // 3  
  
var sum = function(a, b) {  
    var result = a + b;  
  
    return result;  
}  
  
alert( sum(1, 2) ); // 3
```

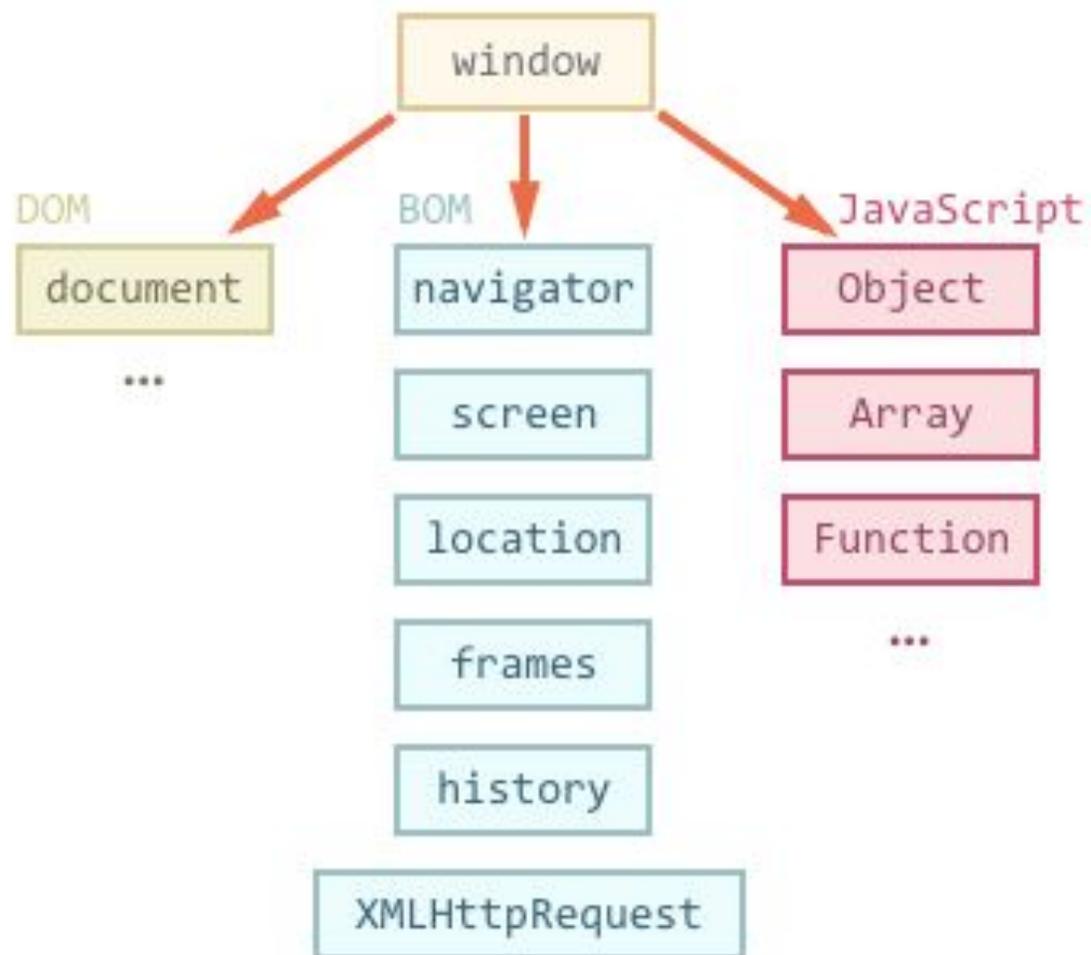


# Объекты

```
person = new Object();  
person = {}; // пустые фигурные скобки  
  
person.name = 'Вася';  
person.age = 25;  
  
alert(person.name + ': ' + person.age); // "Вася: 25"  
delete person.age;  
  
if ("name" in person) {  
    alert("Свойство name существует!");  
}  
  
person['любимый стиль музыки'] = 'Джаз';  
  
var user = {  
    name: "Таня",  
    age: 25,  
    size: {  
        top: 90,  
        middle: 60,  
        bottom: 90  
    }  
}
```



# BOM, DOM, JS



```
window.open('http://ya.ru');
```

```
document.body.style.background = 'red';
```

```
alert( 'Элемент BODY стал красным,' +  
      ' а сейчас обратно вернётся' );
```

```
document.body.style.background = '';
```

```
alert(location.href);
```



# Поиск

## getElementById()

```
<div id="content">Выделим этот элемент</div>

<script>
  var elem = document.getElementById('content');

  elem.style.background = 'red';
</script>
```



## getElementsByTagName()

```
<table id="age-table">
  <tr>
    <td>Ваш возраст:</td>
    <td>
      <label>
        <input type="radio" name="age" value="young" checked> младше 18
      </label>
      <label>
        <input type="radio" name="age" value="mature"> от 18 до 50
      </label>
      <label>
        <input type="radio" name="age" value="senior"> старше 60
      </label>
    </td>
  </tr>
</table>

<script>
  var tableElem = document.getElementById('age-table');
  var elements = tableElem.getElementsByTagName('input');

  for (var i = 0; i < elements.length; i++) {
    var input = elements[i];
    alert( input.value + ': ' + input.checked );
  }
</script>
```



## getElementsByClassName()

```
<div class="article">Статья</div>
<div class="long article">Длинная статья</div>

<script>
  var articles = document.getElementsByClassName('article');
  alert( articles.length ); // 2, найдёт оба элемента
</script>
```



## querySelectorAll() querySelector()

```
<ul>
  <li>Этот</li>
  <li>тест</li>
</ul>
<ul>
  <li>полностью</li>
  <li>пройден</li>
</ul>
<script>
  var elements = document.querySelectorAll('ul > li:last-child');
  for (var i = 0; i < elements.length; i++) {
    alert( elements[i].innerHTML ); // "тест", "пройден"
  }
</script>
```



## Closest()

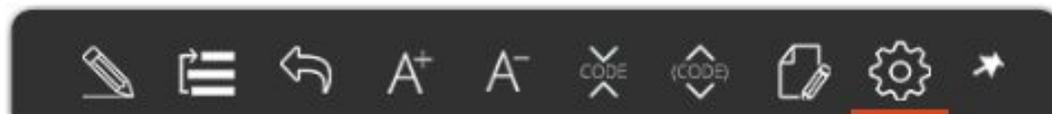
```
<ul>
  <li class="chapter">Глава I
    <ul>
      <li class="subchapter">Глава <span class="num">1.1</span></li>
      <li class="subchapter">Глава <span class="num">1.2</span></li>
    </ul>
  </li>
</ul>

<script>
  var numberSpan = document.querySelector('.num');

  // ближайший элемент сверху подходящий под селектор li
  alert(numberSpan.closest('li').className) // subchapter

  // ближайший элемент сверху подходящий под селектор .chapter
  alert(numberSpan.closest('.chapter').tagName) // LI

  // ближайший элемент сверху, подходящий под селектор span
  // это сам numberSpan, так как поиск включает в себя сам элемент
  alert(numberSpan.closest('span') === numberSpan) // true
</script>
```



# События

## События мыши:

- `click` – происходит, когда кликнули на элемент левой кнопкой мыши
- `contextmenu` – происходит, когда кликнули на элемент правой кнопкой мыши
- `mouseover` – возникает, когда на элемент наводится мышь
- `mousedown` и `mouseup` – когда кнопку мыши нажали или отжали
- `mousemove` – при движении мыши

## События на элементах управления:

- `submit` – посетитель отправил форму `<form>`
- `focus` – посетитель фокусируется на элементе, например нажимает на `<input>`

## Клавиатурные события:

- `keydown` – когда посетитель нажимает клавишу
- `keyup` – когда посетитель отпускает клавишу

## События документа:

- `DOMContentLoaded` – когда HTML загружен и обработан, DOM документа полностью построен и доступен.

## События CSS:

- `transitionend` – когда CSS-анимация завершена.

```
// Способ 1
<input value="Нажми меня" onclick="alert('Клик!')" type="button">
```

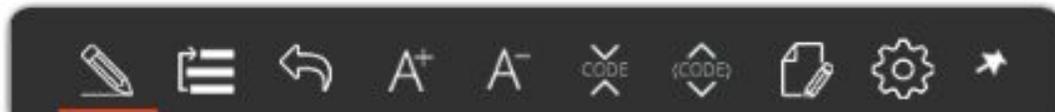
```
// Способ 2
<input id="elem" type="button" value="Нажми меня" />
<script>
  elem.onclick = function() {
    alert( 'Спасибо' );
  };
</script>
```

```
// Способ 3
<input id="elem" type="button" value="Нажми меня"/>

<script>
  function handler1() {
    alert('Спасибо!');
  };

  function handler2() {
    alert('Спасибо ещё раз!');
  }

  elem.addEventListener("click", handler1); // Спасибо!
  elem.addEventListener("click", handler2); // Спасибо ещё раз!
</script>
```



# Задание

1. Нарисовать в HTML таблицу 3 на 6

1	2	3
2	4	6
3	6	9
4	8	12
5	10	15
6	12	18

2. Заполнить ее ячейки с помощью JavaScript таблицей умножения как показано на рисунке
3. Нечетные строки с помощью JavaScript подсветить другим цветом, например как показано на рисунке
4. Для четных строк добавить обработчик события onclick, таким образом, чтобы по нажатию на строчку выводилось Alert-сообщение с содержимым ячейки.

# jQuery

jQuery - библиотека JavaScript, фокусирующаяся на взаимодействии JavaScript и HTML.

Библиотека jQuery помогает легко получать доступ к любому элементу DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими.

Также библиотека jQuery предоставляет удобный API по работе с Ajax.

# Подключение

```
<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.5/jquery.min.js"></script>
```

jQuery() = \$()

```
;(function($) {  
    // Здесь пишем код  
})(jQuery);
```

## Варианты

### ИСПОЛЬЗОВАНИЯ

```
$(<Функция>);  
$(<Элемент объектной модели документа>);  
$(<HTML-текст>);  
$(<Селектор>[, <Контекст>]);
```



# Селекторы

```
$("#myid").css("backgroundColor", "red");  
$(".cls2").css("backgroundColor", "red");  
$("p.cls2").css("backgroundColor", "red");  
$("div > a").css("color", "red");  
$("input[name='myinput']").css("color", "red");
```



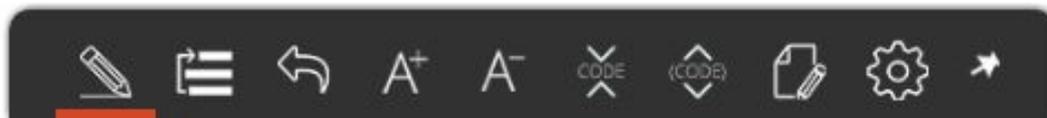
# Атрибуты

```
// вернет имя класса у первого div-элемента на странице  
$("div").attr("class");  
  
// класс всех div-элементов на странице станет равным divEl  
$("div").attr("class", "divEl");  
  
// класс div-элементов станет равен divEl, а title – "Див"  
$("div").attr({"class": "divEl", "title": "Див"});  
  
// вернет подсказку элемента с классом root.  
// Если таких элементов на странице несколько – вернет первого из них.  
$(".root").attr("title");
```



# Значения

```
// вернет значение первого элемента с классом surname  
$(".surname").val();  
  
// вернет значение первого элемента формы с атрибутом name равным surname  
$("input[name='surname']").val();  
  
// присвоит значение "Задерищенко" всем элементам формы с классом surname  
$(".surname").val("Задерищенко");
```



# Работа с CSS

```
// добавит класс blackZone элементу с идентификатором content  
$("#content").addClass("blackZone");  
// добавит класс blackZone всем элементам с классом content  
// То есть, у этих элементов будет два класса: class="content blackZone"  
$(".content").addClass("blackZone");
```

```
$("#p:first").hasClass("selected");
```

```
// удалит все классы у элемента с идентификатором content  
$("#content").removeClass();  
// удалит классы clName1 и clName2 у элементов с классом content  
$(".content").removeClass("clName1 clName2");
```

```
☐ $("#p").click(function () {  
    $(this).toggleClass("highlight");  
});
```



# События

## Стандартные

### события

Click	Mouseleave
Focus	Change
Focusin	Select
Focusout	Submit
Load	Keydown
Resize	Keypress
Scroll	Keyup
Unload	Error

## Специфические

### события

Ready  
Load  
Unload

### Собственные события

```
☐ $('#foo').click(function(){
  alert('Вы нажали на элемент "foo"');
});

☐ $('#foo').bind('click', function(){
  alert('Вы нажали на элемент "foo"');
});

☐ $(document).on("click", "#foo", function() {
  alert('Вы нажали на элемент "foo"');
});
```

```
// установим обработчик
☐ $('#foo').bind('myEvent', function(){
  alert('Шла Саша по шоссе');
});

// вызовем событие
$('#foo').trigger('myEvent');
```



# Ajax

```
$("#result").load("ajax/test.html");
```

```
☐ $.ajax({  
  method: "POST",  
  url: "some.php",  
  data: { name: "John", location: "Boston" },  
  success: function (result) {  
    alert(result);  
  }  
});
```

