# Лабораторная работа № 4

## ОБЪЕКТЫ JAVASCRIPT

Цель работы: ознакомиться с иерархией объектов JAVASCRIPT, способами доступа к объектам на Web-странице.

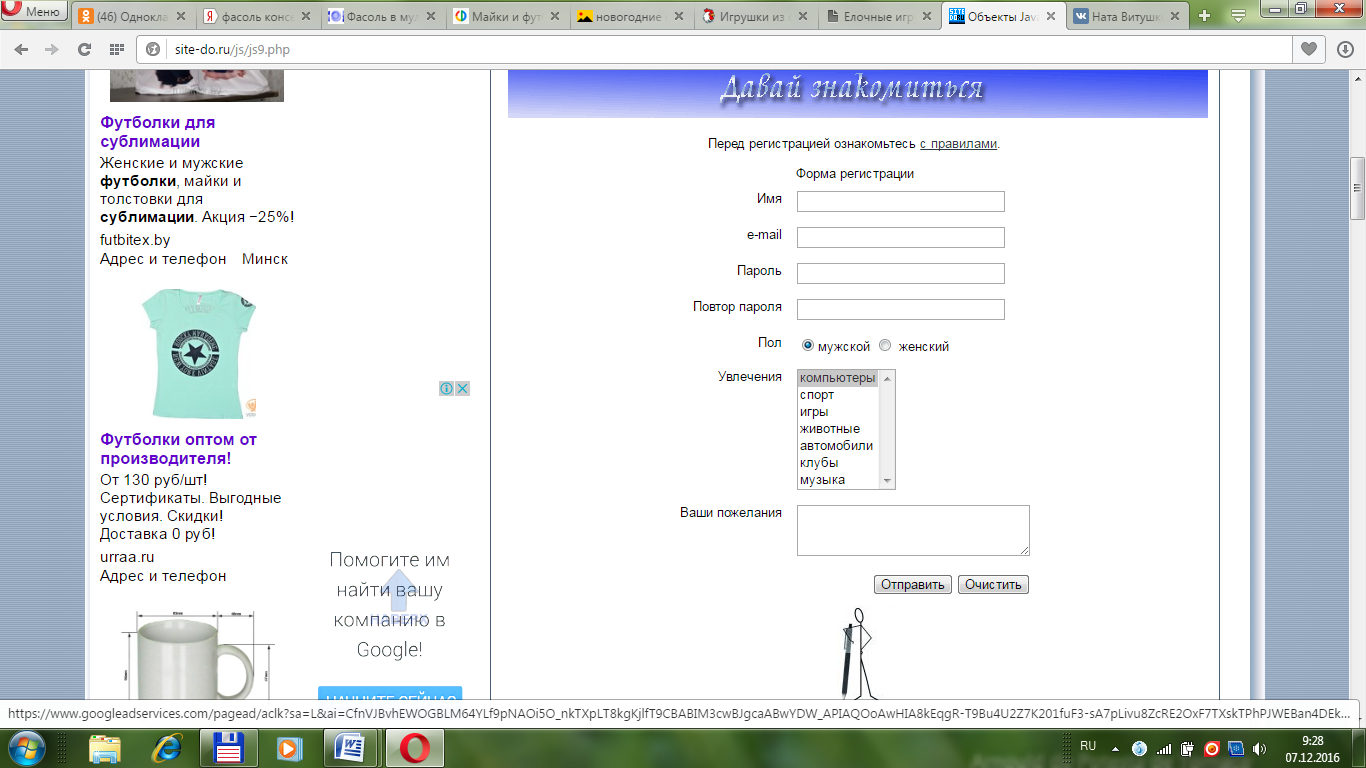
**Задания:**

1. Изучите теоретические сведения.
2. Выполните все примеры, описанные в теоретической части.
3. Оформите отчет о лабораторной работе.

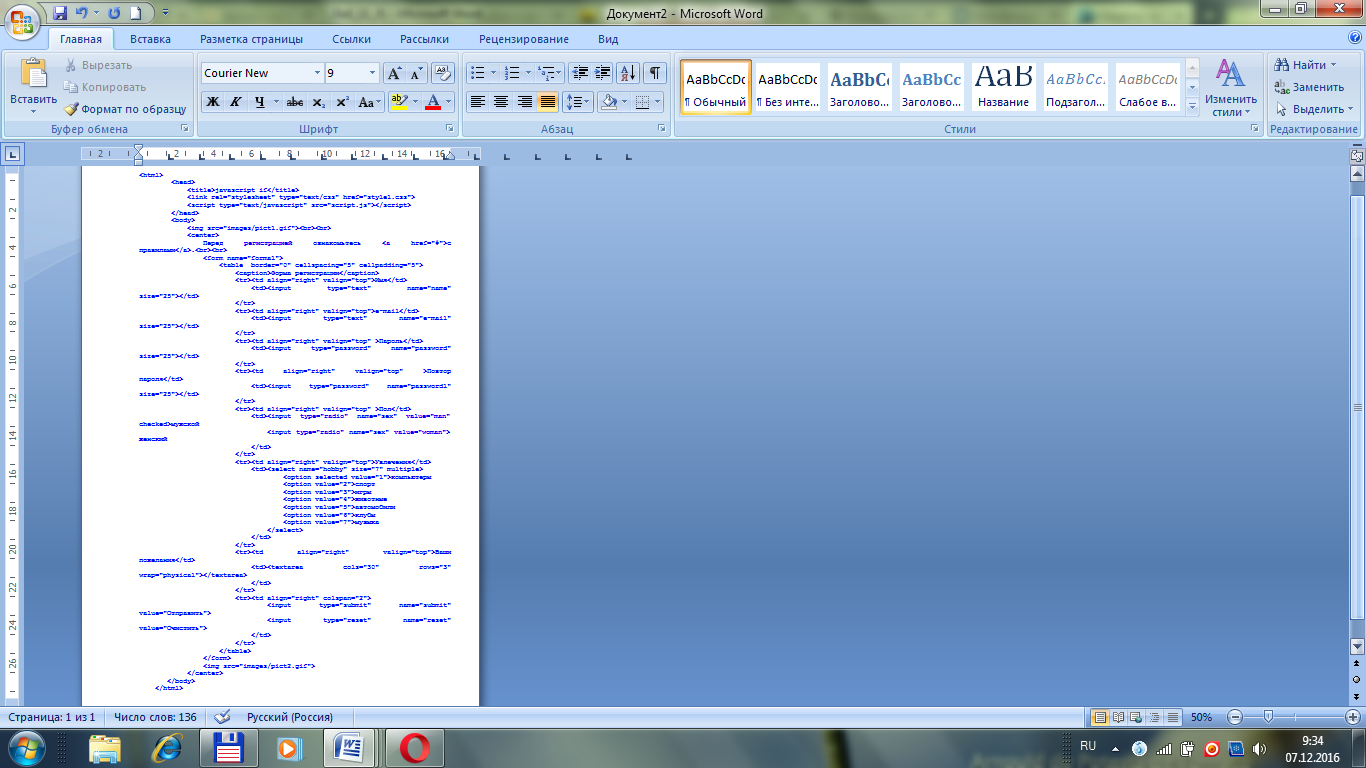
**ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**

В языке JavaScript все элементы на Web-странице выстраиваются в иерархическую структуру. Каждый элемент представляется в виде объекта. И каждый такой объект может иметь определенные свойства и методы.

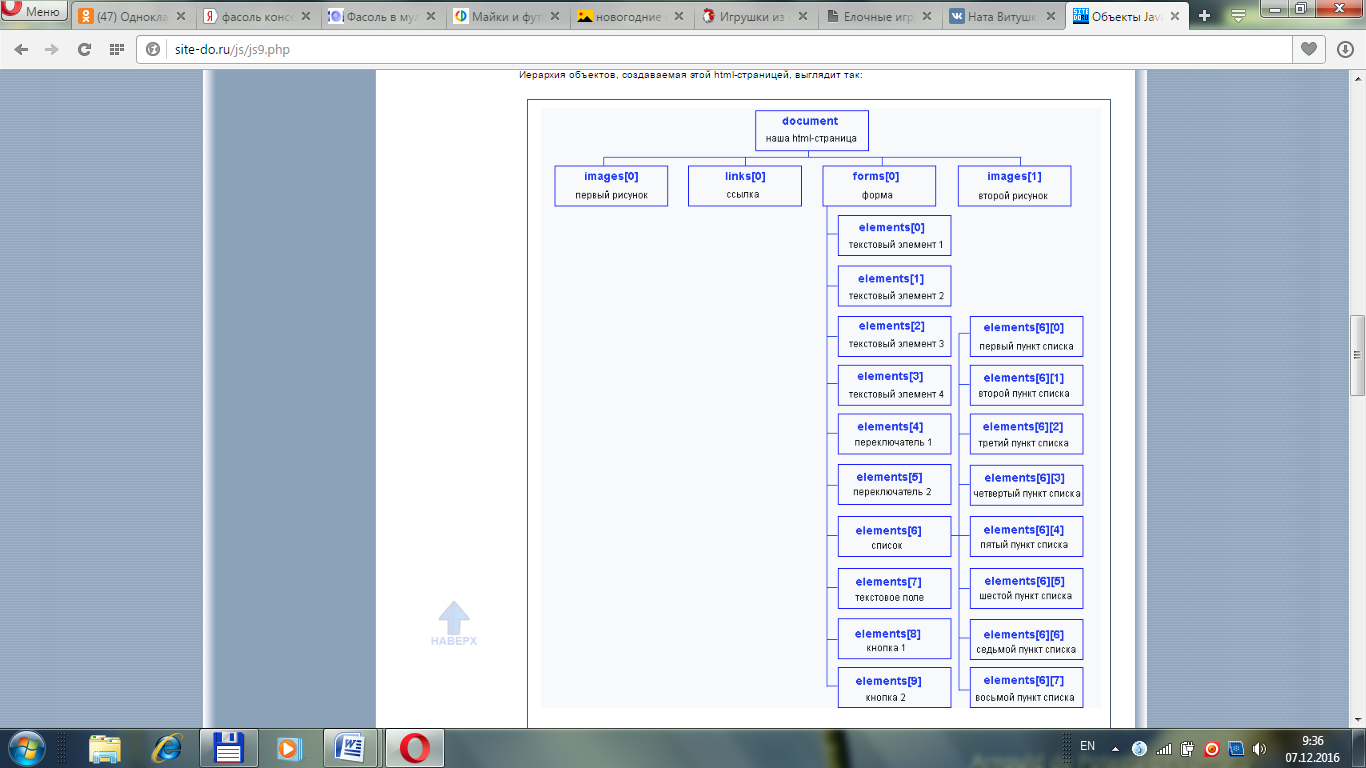
Предположим, у нас есть такая web-страница:



Ее html-код:



Иерархия объектов, создаваемая этой html-страницей, выглядит так:



Каждый объект этой структуры имеет свое имя и индекс. Индекс определяется положением объекта на странице (сверху вниз). Поэтому картинка, которая расположена выше на странице имеет меньший номер. Нумерация начинается с 0. Объект **Form** имеет свойство-массив - **Elements** , в котором содержатся ссылки на элементы формы в порядке их перечисления в теге **Form** . Их нумерация также начинается с 0. У объектов формы могут быть свои подчиненные объекты. В примере у объекта elements[6] который является списком, это пункты списка, которые имеют свою внутреннюю нумерацию.

Чтобы получить доступ к объекту, следует указать весь путь от вершины иерархии до нужного элемента. Так, путь к первой картинке выглядит так:

**document.images[0]**

Путь к форме:

**document.forms[0]**

Путь к списку:

**document.forms[0].elements[6]**

А путь к третьему пункту списка:

**document.forms[0].elements[6][2]**

Такой подход не всегда удобен, так как легко запутаться во всех этих индексах. Поэтому всем элементам на странице обычно даются имена и доступ к элементам осуществляется по именам.

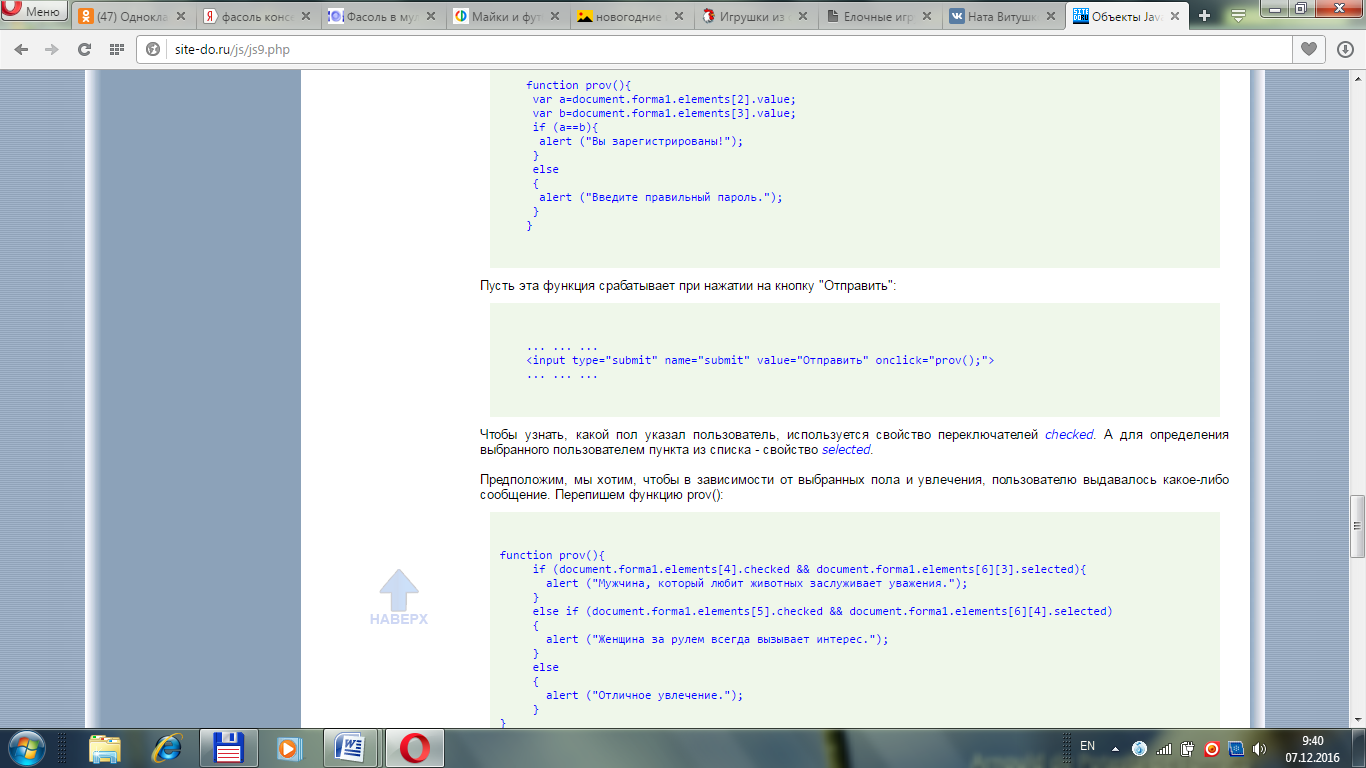
Например, форма имеет имя **Forma1**, а первое текстовое поле - имя **Name**. Тогда, чтобы получить к нему доступ, необходимо написать:

**document.forma1.name**

Чтобы узнать, что пользователь ввел в это текстовое поле, используется свойство **value**:

**document.forma1.name.value**

Предположим, мы хотим осуществлять проверку на совпадение двух полей ввода пароля в нашем примере. Для этого напишем следующую функцию:



Пусть эта функция срабатывает при нажатии на кнопку "Отправить":

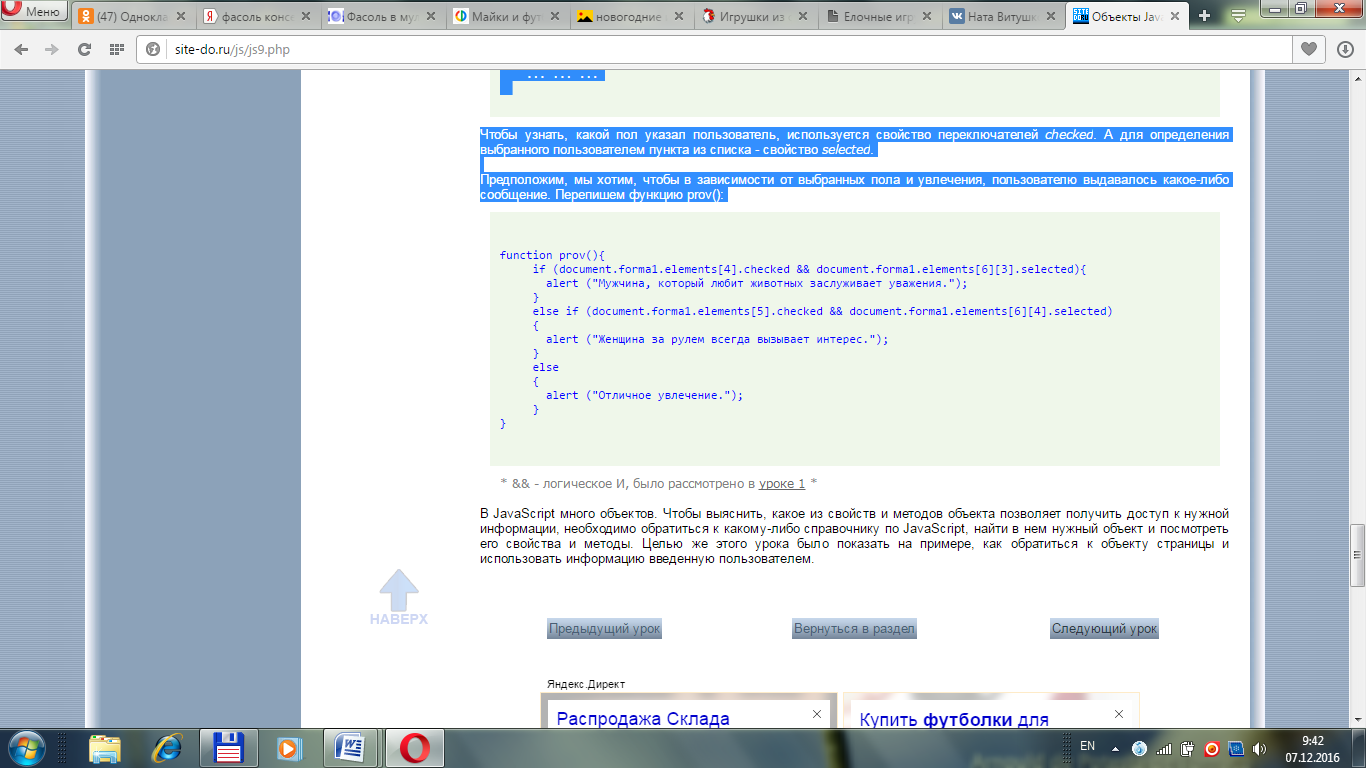
**... ... ...**

**<input type="submit" name="submit" value="Отправить" onclick="prov();">**

**... ... ...**

Чтобы узнать, какой пол указал пользователь, используется свойство переключателей **Checked**. А для определения выбранного пользователем пункта из списка – свойство **Selected**.

Если необходимо, чтобы в зависимости от выбранных пола и увлечения, пользователю выдавалось какое-либо сообщение нужно изменить функцию **prov**():



**Массив**

Массив создается с помощью квадратных скобок **[ ]**:

var arr **=** []; //создаем массив arr

Пока созданный нами массив не содержит никаких данных. **Заполним его** названиями дней недели:

var arr **=** ['пн', 'вт', 'ср', 'чт', 'пт', 'сб', 'вс'];

Каждое значение списка, который мы записали в массив (в нашем случае каждый день недели), называется **элементом** массива.

Элементы разделяются между собой **запятой**. После этой запятой можно ставить пробелы, а можно и не ставить.

Обратите внимание на то, что названия дней недели представляют собой **строки** и поэтому взяты в кавычки. Кроме строк в массиве можно хранить **числа**, и их в кавычки мы не берем:

//В массиве можно хранить как строки, так и числа:

var arr **=** ['пн', 256, 'ср', 34, 38, 'сб', 95];

Предположим, мы хотим вывести на экран среду с помощью составленного массива дней недели.

**Делается это так:** нужно после переменной массива (в нашем случае **arr**) написать квадратные скобки **[ ]**, а в них указать **порядковый номер элемента**, который мы хотим вывести: **arr[3]**. Казалось бы, что порядковый номер среды - это 3, но это не так. Потому что в программировании нумерация начинается с нуля. Поэтому 0 - это понедельник, 1 - это вторник, а 2 - это среда.

Посмотрите и повторите пример:

//Выведем слово 'ср':

var arr **=** ['пн', 'вт', 'ср', 'чт', 'пт', 'сб', 'вс'];

alert(arr[2]);

В предыдущем примере, чтобы обратиться к нужному элементу массива, мы писали в квадратных скобках его **порядковый номер** (нумерация начинается с нуля, если вы уже забыли). Эти порядковые номера называются **ключами массива**.

То есть мы получали *значение элемента массива по его ключу*.

В нашем случае JavaScript **сам определял ключи** для элементов - это были их **порядковые номера**. Иногда это может оказаться **неудобным** - к примеру, если мы хотим вывести на экран название третьего дня недели (среду), то должны написать в квадратных скобках цифру 2, а не 3.

Поэтому в JavaScript можно указать ключи в **явном** виде – так, как нам нужно. Делается это с помощью **объектов** (в других языках программирования они называются **ассоциативными** массивами). Объекты создаются с помощью фигурных скобок **{ }**, внутри которых пишутся элементы этого объекта в формате **ключ: значение**.

Давайте сделаем так, чтобы **понедельник** имел **ключ 1**, а не ноль, как было раньше (и всем остальным дням прибавим единицу):

//Укажем ключи в явном виде:

var obj **=** {1: 'пн', 2: 'вт', 3: 'ср', 4: 'чт', 5: 'пт', 6: 'сб', 7: 'вс'};

alert(obj[1]); //выведет 'пн'

Синтаксис здесь такой: **ключ**, затем идет двоеточие **:**, а потом **значение**.

Ключи не обязательно должны быть числами, они могут быть и **строками**. Сделаем массив, в котором ключами будут имена работников, а элементами - их зарплаты:

//Массив (объект) работников:

var obj **=** {'Коля': 200, 'Вася': 300, 'Петя': 400};

Узнаем зарплату Васи:

var obj **=** {'Коля': 200, 'Вася': 300, 'Петя': 400};

alert(obj['Вася']); //выведет 300

Кроме того, кавычки вокруг строковых ключей можно и не ставить:

var obj **=** {key1: 200, key2: 300, key2: 400};

alert(obj['key1']); //выведет 200

Однако, на такие ключи накладываются ограничения: они не должны начинаться с цифры, не должны иметь символ дефиса или символ пробела внутри.

Если у вас есть такой ключ - просто возьмите его в кавычки и все будет ок.

**Свойства объекта**

Существует и другой способ обратиться к элементу объекта - используя обращение не через квадратные скобки, а через точку: не **obj['key']**, а **obj.key**.

В этом случае говорят, что мы обращаемся к *свойству* объекта.

Смотрите пример:

var obj **=** {key1: 200, key2: 300, key2: 400};

alert(obj.key1); //выведет 200

При таком способе обращения на названия свойств (то есть ключей) накладываются ограничения: они не должны начинаться с цифры, не должны иметь символ дефиса или символ пробела внутри.

Если у вас есть такой ключ - используйте обращение через квадратные скобки.

**Как еще можно создать массив или объект**

Вместо **{ }** и **[ ]** можно пользоваться **Object()** и **Array()** соответственно (это просто альтернативный синтаксис).

Давайте создадим объект вторым синтаксисом:

var obj **=** new Object(key1: 200, key2: 300, key2: 400);

alert(obj['key1']); //выведет 200

Давайте создадим массив вторым синтаксисом:

var arr **=** Array('пн', 256, 'ср', 34, 38, 'сб', 95);

Создание объекта через **{ }** и через **Object()** эквивалентно, вы можете пользоваться тем способом, который вам удобнее.

А вот с массивом не все так просто. Разница между ними проявляется в случае, если вы хотите сделать массив, состоящий из одного элемента, который будет целым числом, вот так:

var arr **=** Array(10);

В этом случае получим не такой массив:

var arr **=** [10];

А вот такой:

var arr **=** [,,,,,,,,,];

Это будет массив, состоящий из 10-ти пустых элементов (их значение будет undefined), а не массив из одного элемента 10, как мы хотели бы.

**Заполнение массива**

Массив можно заполнять не на этапе его создания, а потом. В следующем примере я вначале объявил, что переменная **arr** - массив, а потом заполнил его данными:

var arr **=** []

arr[0] **=** 1;

arr[1] **=** 2;

arr[2] **=** 3;

alert(arr) //с помощью alert выводим содержимое массива

Также можно поступать и с объектами:

var obj **=** {};

obj['Коля'] **=** 100;

obj['Вася'] **=** 200;

obj['Петя'] **=** 300;

**Многомерный массив**

Элементы массива могут быть не только строками и числами, но и массивами. То есть у нас получится **массив массивов** или **многомерный массив**.

Давайте сделаем массив студентов **students**, который будет содержать два подмассива: студенты мужского пола и женского:

//Многомерный массив студентов:

var students **=** {

'boys': ['Коля', 'Вася', 'Петя'],

'girls': ['Даша', 'Маша', 'Лена'],

};

Чтобы вывести какой-либо элемент из многомерного массива следует писать уже не одну пару **[ ]**, а две: **students['boys'][0]** – так мы выведем 'Коля'.

**Примеры решения задач.**

**Задача 1.** Дан массив с элементами 'Привет, ', 'мир' и '!'. Необходимо вывести на экран фразу 'Привет, мир!'.

**Решение:**

//Выведем фразу 'Привет, мир!':

var arr **=** ['Привет, ', 'мир', '!'];

alert(arr[0] **+** arr[1] **+** arr[2]);

Разберем это решение: слово 'Привет, ' хранится под номером **0**, это значит, что для доступа к нему мы должны написать **arr[0]**. Для доступа к слову 'мир' мы должны написать **arr[1]**, а **arr[2]** содержит в себе '!'. Далее с помощью оператора **'+'** мы сложим три наши строки ('Привет, ', 'мир' и '!') в одну строку таким образом **arr[0] + arr[1] + arr[2]**, и выведем на экран с помощью **alert**.

**Задача 2.** Решим немного другую задачу: дан массив с элементами 'Привет, ', 'мир' и '!'. Необходимо записать в переменную **text** фразу 'Привет, мир!', а затем вывести на экран содержимое этой переменной.

**Решение:**

//Выведем фразу 'Привет, мир!':

var arr **=** ['Привет, ', 'мир', '!'];

var str **=** arr[0] **+** arr[1] **+** arr[2]; //в переменной text теперь лежит строка 'Привет, мир!'

alert(str);

**Задача 3.** Дан массив ['Привет, ', 'мир', '!']. Необходимо записать в нулевой элемент **этого** массива слово 'Пока, ' (то есть вместо слова **'Привет, '** будет **'Пока, '**).

**Решение:**

var arr **=** ['Привет, ', 'мир', '!'];

arr[0] **=** 'Пока, '; //перезапишем нулевой элемент массива

alert(arr); //посмотрим на массив и убедимся в том, что он изменился

**Задача 4.** Создайте ассоциативный массив (объект) заработных плат **obj**. Выведите на экран зарплату Пети и Коли.

//Этот объект дан:

var obj **=** {'Коля':'1000', 'Вася':'500', 'Петя':'200'};

**Решение:** Чтобы вывести зарплату Коли следует вывести значение элемента объекта **с ключом 'Коля'**. Сделаем это:

var obj **=** {'Коля':'1000', 'Вася':'500', 'Петя':'200'};

alert(obj['Коля']); //выведет 1000

**Задача 5.** Создайте массив **arr** с элементами *1, 2, 3, 4, 5* двумя различными способами.

**Решение:**

Первый способ создать массив - объявить его через **[ ]**:

var arr **=** [1, 2, 3, 4, 5];

Второй способ создания массива - это поступить таким образом:

var arr **=** [];

arr[0] **=** 1;

arr[1] **=** 2;

arr[2] **=** 3;

arr[3] **=** 4;

arr[4] **=** 5;

**Задача 6.** Дан многомерный массив **arr**:

var arr **=** {

'ru':['голубой', 'красный', 'зеленый'],

'en':['blue', 'red', 'green'],

};

Выведите с его помощью слово **'голубой'**.

**Решение:** так как массив многомерный (в нашем случае двухмерный), то нам придется написать несколько квадратных скобок подряд. Поясню это по шагам. Давайте сделаем так:

alert(arr['ru']);

В этом случае результатом будет массив **['голубой','красный', 'зеленый']**, который является частью нашего многомерного массива. Чтобы вывести слово **'голубой'**, необходимо дописать еще одну квадратную скобку с ключом, соответствующим этому элементу (у него нет явного ключа - значит его ключ **0**):

alert(arr['ru'][0]); //выведет 'голубой'

Выведем теперь слово **'красный'**:

alert(arr['ru'][1]); //выведет 'красный'

Выведем **'red'**:

alert(arr['en'][1]); //выведет 'red'

**Задачи для самостоятельного решения**

По вариантам (4 варианта, 3 задания)

**Вариант1**

1. Создайте массив **arr = ['a', 'b', 'c']**. Выведите его на экран с помощью функции **alert**.
2. С помощью массива **arr** из предыдущего номера выведите на экран содержимое **первого**, **второго** и **третьего** элементов.
3. Создайте массив **arr = ['a', 'b', 'c', 'd']** и с его помощью выведите на экран строку **'a+b, c+d'**.
4. Создайте массив **arr** с элементами **2, 5, 3, 9**. Умножьте первый элемент массива на второй, а третий элемент на четвертый. Результаты сложите, присвойте переменной **result**. Выведите на экран значение этой переменной.

**Вариант2**

1. Создайте объект **obj**. Выведите на экран элемент с ключом **'c'** двумя способами: через квадратные скобки и как свойство объекта:

var obj **=** {a: 1, b: 2, c: 3};

1. Создайте массив заработных плат **obj**. Выведите на экран зарплату Пети и Коли.

var obj **=** {Коля: '1000', Вася: '500', Петя: '200'};

1. Создайте объект с днями недели. Ключами в нем должны служить номера дней от начала недели (понедельник - первый и т.д.). Выведите на экран **текущий** день недели.
2. Пусть теперь номер дня недели хранится в переменной **day**, например там лежит число 3. Выведите день недели, соответствующий значению переменной **day**.

**Вариант3**

1. Дан массив **[ [1, 2, 3], [4, 5, 6], [7,8,9] ]**. Выведите на экран цифру **4** из этого массива.
2. Дан объект **{js:['jQuery', 'Angular'], php: 'hello', css: 'world'}**. Выведите с его помощью слово **'jQuery'**.
3. Создайте двухмерный массив. Первые два ключа - это **'ru'** и **'en'**. Пусть первый ключ содержит элемент, являющийся массивом названий дней недели *по-русски*, а второй - *по-английски*. Выведите с помощью этого массива понедельник по-русски и среду по английски (пусть понедельник - это нулевой день).
4. Пусть теперь в переменной **lang** хранится язык (она принимает одно из значений или 'ru', или 'en' - либо то, либо то), а в переменной **day** - номер дня. Выведите словом день недели, соответствующий переменным **lang** и **day**. То есть: если, к примеру, lang = 'ru' и day = 3 - то выведем 'среда'.

**КОНТРОЛЬНЫЕ ВОПРОСЫ:**

1. В виде какой структуры можно представить все элементы все на Web-странице в языке JavaScript?
2. Назовите характеристики объектов Web-страницы.
3. Как определяется индекс объекта?
4. Какой массив элементов содержит объект **Form**? Как нумеруются элементы объекта **Form**?
5. Как получить доступ к объекту? Приведите пример.
6. Для чего объектам на страница даются имена? Приведите пример обращения к объектам по имени.
7. Для чего используется свойство **Value?**