**Основы работы с массивами и   
объектами в JavaScript**

В программировании очень часто возникает задача **хранения списка похожих значений**, например, всех дней недели или всех месяцев. Можно было бы создавать под каждое значение списка свою переменную, но это очень неудобно и долго - для списка дней недели понадобилось бы 7 переменных, а для месяцев - вообще 12.

А теперь представьте, что вам нужно вывести на экран название третьего месяца. С помощью 12-ти различных переменных это было бы проблематично, так как вам необходимо помнить названия всех переменных, в которые вы записали имена месяцев.

Поэтому для таких вещей был изобретен специальный тип данных. Он называется **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];

## Как вывести отдельный элемент массива

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]);

## Ассоциативные массивы (объекты)

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'

<http://code.mu/books/javascript/base/osnovy-raboty-s-massivami-i-objektami-v-javascript.html>