Объекты JavaScript

Объект – это составной тип данных, он объединяет множество значений в единый модуль и позволяет сохранять и извлекать значения по их именам. **Объекты** – это неупорядоченные коллекции свойств, каждое из которых имеет свое имя и значение.

В качестве свойств могут быть обычные переменные, содержащие значения, или объекты.

Создание объектов

Способы создания объектов:

1. с помощью оператора **new** за которым следует функция конструктор;
2. с помощью **литерала объекта**.

1. var empty = **new** Object(); //пустой объект создается с помощью оператора new

2. var empty = {}; //пустой объект создается с помощью литерала объекта

Литерал объекта

Самым простым способом создания и инициализации объекта, является использование литерала объекта.

Литерал объекта – это заключенный в фигурные скобки список свойств, разделенных запятыми. Имя каждого свойства может быть обычным идентификатором (как имя переменной) или строкой, а значением любого свойства может быть обычное значение (строка число и т.д.) или другой объект.

Пример

1. var empty = {}; //создали пустой объект
2. var point = {x:0, y:0}; //объект с двумя свойствами x и y со значениями 0
3. var homer = { //объект с несколькими свойствами
   "name": "Гомер Симпсон",
   age: 34,
   married: true
};

После фигурной скобки должна стоять точка с запятой.

Для доступа к значениям свойств объекта используется оператор точка **. (точка).**

Значение в левой части оператора должно быть ссылкой на объект, к свойствам которого требуется получить доступ.

Значение в правой части оператора должно быть именем свойства. Свойства объекта работают как переменные: в них можно сохранять значения и считывать их.

Пример

```
var homer = {
  name: "Гомер Симпсон",
  age: 34,
};
  homer.age = 35;     //присваиваем свойству новое значение
document.write(homer.age + "<br>");  //вывод значения свойства age
 homer.male = "мужчина";  //добавляем в объект новое свойство со значением
document.write(homer.male); //вывод значения нового свойства объекта
```

Новое свойство объекта, можно добавить в любом месте, присвоив этому свойству значение, так же и значения свойств уже созданных, можно изменять в любой момент простым присваиванием нового значения.

Значения свойств указанных внутри литерала объекта указываются после двоеточия, если добавляется новое свойство или присваивается новое значение уже существующему за пределами литерала объекта, то вместо двоеточия используется операция присваивания.

Пример

```
var obj = {
  name: "Гомер",
  colors: {
    first: "yellow",
    second: "blue"
  }
};

//для доступа к значениям свойств используется стандартный синтаксис
document.write(obj.colors.first);
```

Значением свойства **colors** является объект **{first: "yellow", second: blue}**.

У объекта (который выступает в качестве значения) в конце отсутствует точка с запятой.

Проверка, перечисление и удаление свойств

Для удаления свойств объекта используется оператор **delete**.

```
var homer = {
  name: "Гомер Симпсон",
  age: 34,
  married: true
};

delete homer.age;
```

Для проверки факта существования того или иного свойства используется оператор **in.**

С левой стороны от оператора помещается имя свойства в виде строки, с правой стороны - проверяемый объект на наличие указанного свойства. При обращении к несуществующему свойству возвращается значение undefined.

```
var obj = {};
if ("a" in obj) {
  obj.a = 1;
}
else {
  document.write("Такого свойства не существует");
}
```

Доступа к свойству через квадратные скобки []

Доступ к свойствам объекта возможен также при помощи оператора [], который обычно используется при работе с массивами.

```
obj.property = 10;
obj['property'] = 10;
```

Различие между этими двумя синтаксисами:

в первом варианте имя свойства представляет собой идентификатор, а во втором - строку.

Методы объекта

Метод – это функция, которая хранится в качестве значения в свойстве объекта и может вызываться посредством этого объекта.

```
var obj = {
  name: "Гомер",
     write_hello: function() {
    document.write("Привет");
  }
};
 obj.write_hello();  //вызов метода
```

Для **доступа** к объекту из метода используется ключевое слово **this**. Оно ссылается на объект, в контексте которого вызван метод и позволяет обращаться к другим его методам и свойствам.

```
var calc = {
  num1: 5,
  num2: 5,
  compute: function( ) {
    this.result = this.num1 * this.num2;
  }
};
 calc.compute(); //Вычисляем сколько будет 5*5?
document.write(calc.result); // Выводим результат
```

Функция конструктор и оператор new

Конструктор – это функция, которая выполняет инициализацию свойств объекта и предназначена для использования совместно с оператором new.

```
//определяем конструктор
function Car(seats){
  this.seats = seats;
  this.canDrive = true;
}
//вызываем конструктор для создания нового объекта
var myCar = new Car("leather");
```

Оператор **new** создает новый пустой объект без каких-либо свойств, а затем вызывает функцию-конструктор, передавая ей только что созданный объект.

Главная задача конструктора заключается в инициализации вновь созданного объекта - установке всех его свойств, которые необходимо инициализировать до того, как объект сможет использоваться программой. После того как объект создан и инициализирован, переменной myCar присваивается ссылка на объект.

Результатом выполнения кода из примера является создание нового экземпляра объекта:

```
myCar = {
  seats: "leather",
  canDrive: true
};
```

Создаваемые объекты таким образом обычно называют **экземпляром объекта (или класса),** в нашем случае **myCar** является экземпляром объекта **Car**.

При вызове конструктора без аргументов, скобки можно не ставить.

```
var myCar = new Car;
//тоже самое, что и
var myCar = new Car();
```