**Объект Object()**

Объект - это конструкция, которая может иметь свойства и методы. Кроме стандартных объектов языка JavaScript, от программистов требуется умение создавать собственных объекты, которые отсутствуют в языке, но помогают решать ряд задач.

**В JavaScript есть три категории объектов:**

1. Объекты базового типа – это объекты, определённые в спецификации ECMAScript. Например, объекты типа Array, Function, Date или RegExp являются объектами базового типа.
2. Объекты среды выполнения – это объекты, определённые в среде выполнения (такой как браузер). Например, объекты типа document, являются объектами среды выполнения. Создавать новые объекты этой категории и вносить изменения в существующие, могут только разработчики браузеров, в которых и будет интерпретироваться код JavaScript.
3. Пользовательские объекты – это объекты, созданные программистами (front-end и back-end) в результате выполнения программного кода JavaScript. Такие объекты значительно расширяют язык JavaScrip. На их основе создаются пользовательские библиотеки. Такие объекты по важности и приоритетам, можно прировнять к первой группе – объектам базового типа.

Объекты в JavaScript, как и во многих других языках программирования, похожи на объекты реальной жизни. Концепцию объектов JavaScript легче понять, проводя паралелли с реально существующими в жизни объектами.

В JavaScript пользовательский объект — это самостоятельная единица, имеющая свойства и определенный тип. Сравним, например, с чашкой. У чашки есть цвет, форма, вес, материал, из которого она сделана, и т.д. Точно так же, объекты JavaScript имеют свойства, которые определяют их характеристики.

В сравнении с объектами базового типа, пользовательские объекты достаточно миниатюрны и явно разглядеть их работу на каком-то сайте, достаточно сложная задача. Допустим, вы создаете сайт на котором будет продаваться упомянутая чашка. С помощью JavaScript, вы моделируете ее свойства: внешний вид (фотография или даже несколько), доступные цвета, материал из которого она сделана, цену, ее назначение (текстовое описание) и оборачиваем все это скриптами в объект. А теперь представьте, что на одной странице требуется разместить не одну чашку, а штук 20. По аналогии с объектами базового типа, создаем 20 экземпляров нашего объекта. А если таких чашек на сайте 1000? И кроме чашек есть еще тарелки, блюдца, ложки и тд? Где же хранить все эти экземпляры?

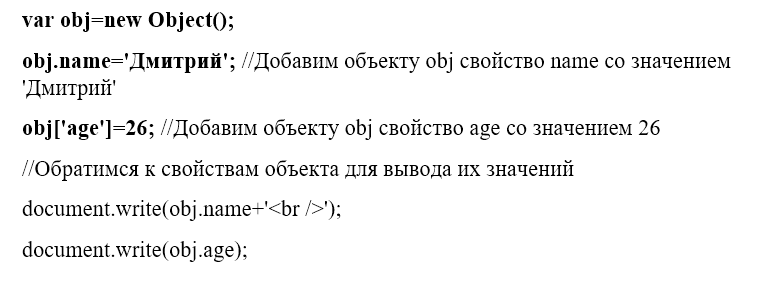
В оперативной памяти. А как туда что-то поместить? Переменные.

В JavaScript объект имеет свойства, ассоциированные с ним. Свойство объекта можно понимать как переменную, закрепленную за объектом. Свойства объекта в сущности являются теми же самыми переменными JavaScript, за тем исключением, что они закреплены за объектом. Свойства объекта определяют его характеристики. Получить доступ к свойству объекта можно так:

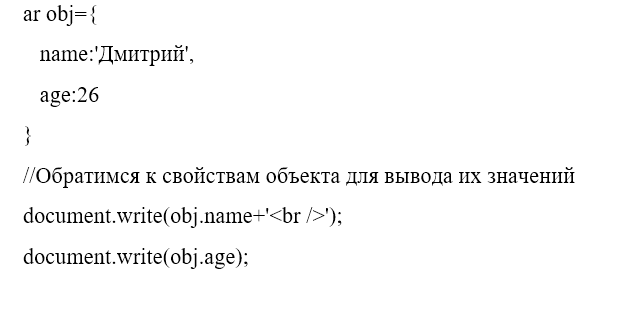
**objectName.propertyName**

Таким образом, Объекты являются **ассоциативными массивами**, так как свойст может быть несколько и каждое ассоциировано с именем объекта, через которое можно получить доступ к нему.

**Объекты могут быть созданы двумя способами:**



**второй способ (упрощенный. По аналогии с Array)**



В JavaScript можно обращаться к свойствам объектов двумя способами:

Используя точку ('.') после имени объекта:

//Присваиваем свойству объекта произвольное значение

имя\_объекта.свойство=значение

//Обращаемся к значению свойства объекта

имя\_объекта.свойство

Заключая название свойства в квадратные скобки ([]) после имени объекта:

//Присваиваем свойству произвольное значение

имя\_объекта['свойство']=значение

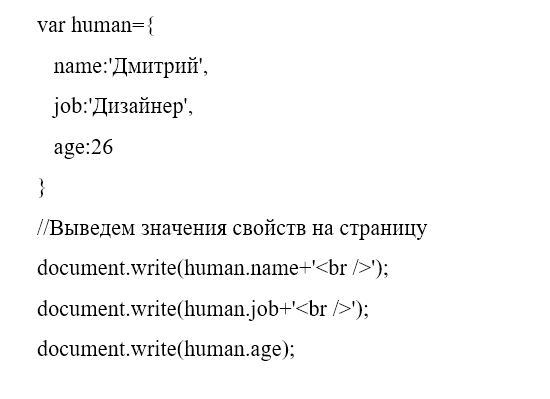
//Обращаемся к значению свойства

имя\_объекта['свойство']

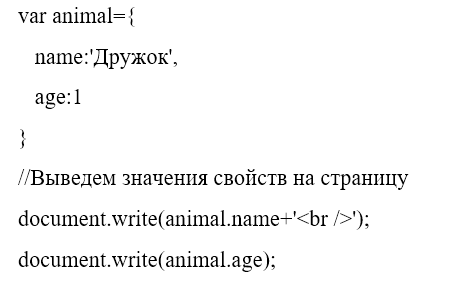
Первый вариант, самый распространенный, поскольку исключает путаницу между ассоциативными массивами (объектами) и упорядоченными массивами (Array).

**Свойства** - это значения, связанные только с текущим объектом и/или его экземплярами.

Например, объект человек может иметь следующие свойства: имя, возраст, профессия.

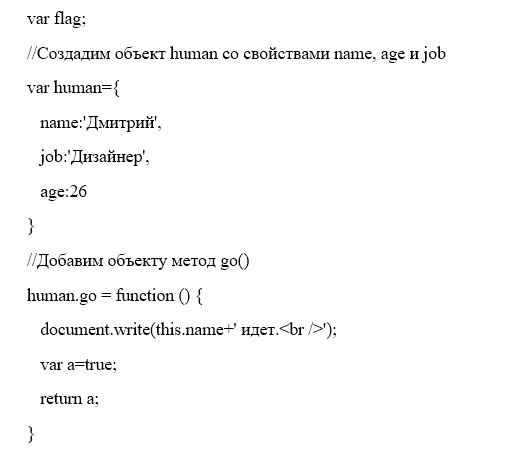


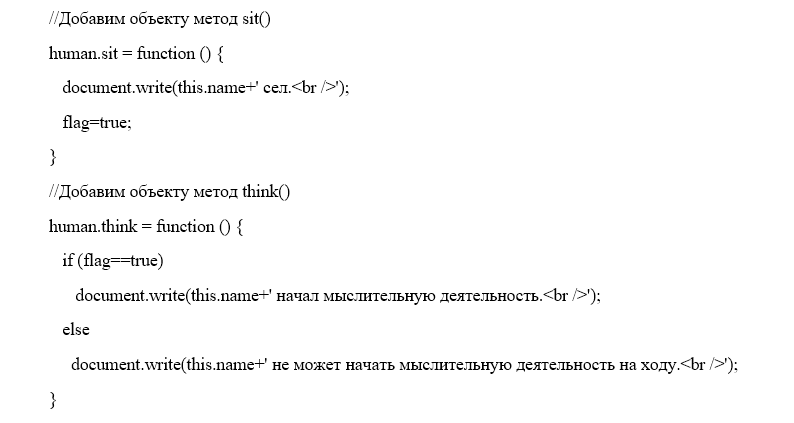
Свойства могут иметь одинаковые названия, но при этом относиться к совершенно разным объектам.

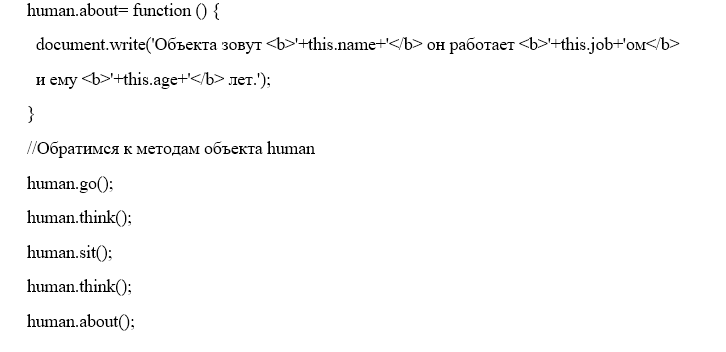


**Методы** – это функции, связанные с текущим объектом и/или его экземплярами. Без объекта или с другими объектами(не экземплярами) они не работают.

Например, объект человек может иметь следующие методы: идти(), сидеть(), думать().

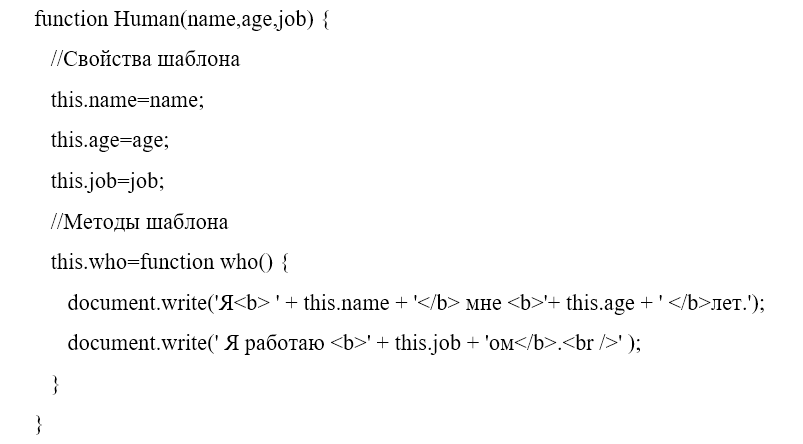


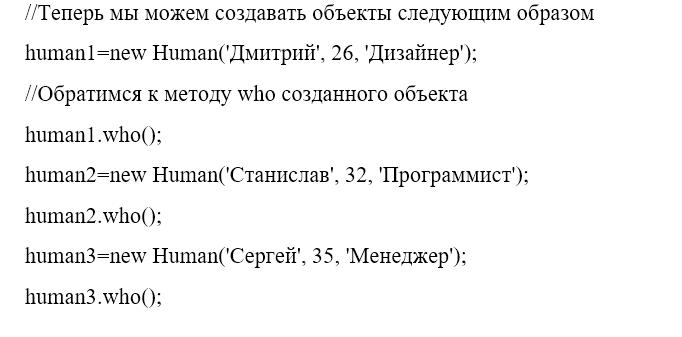




**Конструктор объектов** - это шаблон, на основе которого создаются экземпляры объекта. При создании шаблона, тело ассоциативного массива (объекта) оборачивается в функцию, к которой будем обращаться в момент создания экземпляров пользовательского объекта. Такие шаблоны, можно выносить в отдельные файлы и использовать в качестве библиотек в однотипных проектах.

//Создадим конструктор, который будет создавать объекты со свойствами name, age,job и методом who





**Обратите внимание:** ключевое слово **this** используется для обозначения "текущего" объекта. Это контекстный указатель на объект. К примеру при создании объекта human1, this в конструкторе Human подменяется на human1, а при создании объекта human2 подменяется на human2. This активно используется при работе с пользовательскими объектами, чтобы исключить путаницу в свойствах и методах разных объектов.

Оператор **new** обязательно должен присутствовать при создании экземпляра пользовательского объекта.

**Задание:**

Создайте объект Чашка и несколько его экземпляров.