Лабораторная работа №16

Основные свойства JavaScript

- В JavaScript объект имеет свойства, ассоциированные с ним. Свойство объекта можно понимать как переменную, закреплённую 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 с помощью точечной записи:

- objectName.propertyName

- Как и все переменные JavaScript, имя объекта (которое тоже может быть переменной) и имя свойства являются чувствительными к регистру. Вы можете определить свойство указав его значение. Например, давайте создадим объект myCar и определим его свойства make, model, и year следующим образом:

- var myCar = new Object();
- myCar.make = "Ford";
- myCar.model = "Mustang";
- myCar.year = 1969;
- Неопределённые свойства объекта являются undefined (а не null).
- myCar.color; // undefined

- Свойства объектов JavaScript также могут быть доступны или заданы с использованием скобочной записи (более подробно см. property accessors). Объекты иногда называются *ассоциативными массивами*, поскольку каждое свойство связано со строковым значением, которое можно использовать для доступа к нему. Так, например, вы можете получить доступ к свойствам объекта myCar следующим образом:

- myCar["make"] = "Ford";
- myCar["model"] = "Mustang";
- myCar["year"] = 1969;

- Имена свойств объекта могут быть строками JavaScript, или тем, что может быть сконвертировано в строку, включая пустую строку. Как бы то ни было, доступ к любому имени свойства, которое содержит невалидный JavaScript идентификатор (например, имя свойства содержит в себе пробел и тире или начинается с цифры), может быть получен с использованием квадратных скобок. Этот способ записи также полезен, когда имена свойств должны быть динамически определены (когда имя свойства не определено до момента исполнения). Примеры далее:

```
var myObj = new Object(),
    str = "myString",
    rand = Math.random(),
    obj = new Object();

myObj.type              = "Dot syntax";
myObj["date created"]   = "String with space";
myObj[str]              = "String value";
myObj[rand]             = "Random Number";
myObj[obj]              = "Object";
myObj[""]               = "Even an empty string";

console.log(myObj);
```

- Обратите внимание, что все ключи с квадратными скобками преобразуются в тип String, поскольку объекты в JavaScript могут иметь в качестве ключа только тип String. Например, в приведённом выше коде, когда ключ obj добавляется в myObj, JavaScript вызывает метод obj.toString () и использует эту результирующую строку в качестве нового ключа.
- Вы также можете получить доступ к свойствам, используя значение строки, которое хранится в переменной:

- var propertyName = "make";
- myCar[propertyName] = "Ford";
-
- propertyName = "model";
- myCar[propertyName] = "Mustang";

- Вы можете пользоваться квадратными скобками в конструкции for...in чтобы выполнить итерацию всех свойств объекта, для которых она разрешена. Чтобы показать как это работает, следующая функция показывает все свойства объекта, когда вы передаёте в неё сам объект и его имя как аргументы функции:

- function showProps(obj, objName) {
- var result = "";
- for (var i in obj) {
- if (obj.hasOwnProperty(i)) {
- result += objName + "." + i + " = " + obj[i] + "\n";
- }
- }
- return result;
- }
- Так что если вызвать эту функцию вот так showProps(myCar, "myCar"), то получим результат:
- myCar.make = Ford
- myCar.model = Mustang
- myCar.year = 1969

- Определение свойств для типа объекта
- Вы можете добавить свойство к ранее определённому типу объекта воспользовавшись специальным свойством prototype. Через prototype создаётся свойство, единое для всех объектов данного типа, а не одного экземпляра этого типа объекта. Следующий код демонстрирует это, добавляя свойство color ко всем объектам типа car, а затем присваивая значение свойству color объекта car1.
- Car.prototype.color = null;
- car1.color = "black";
- Смотрите свойство prototype (en-US) объекта Function в Справочнике JavaScript для получения деталей.

Удаление свойств

- Вы можете удалить свойство используя оператор delete. Следующий код показывает как удалить свойство.
- //Creates a new object, myobj, with two properties, a and b.
- var myobj = new Object;
- myobj.a = 5;
- myobj.b = 12;
-
- //Removes the a property, leaving myobj with only the b property.
- delete myobj.a;
- Вы также можете воспользоваться delete чтобы удалить глобальную переменную, если ключевое слово var не было использовано при её объявлении:
- g = 17;
- delete g;
- Смотри delete чтобы получить дополнительную информацию.
-