Javascript Objects

- description : Javascript Object Definitions
- author : 오션
- email : shlim@repia.com
- lastupdate : 2021-04-29

The source of this article

Javascript Object Definitions
JavaScript에서, 오브젝트(objects, 객체)는 King입니다. 오브젝트를 이해하면, JavaScript를 이해할 수 있습니다.

JavaScript에서 거의 " 모든 것 " 이 오브젝트입니다.

- Booleans는 오브젝트가 될 수 있습니다 (new 키워드로 정의된 경우)
- Numbers는 오브젝트가 될 수 있습니다 (new 키워드로 정의된 경우)
- Strings은 오브젝트가 될 수 있습니다 (new 키워드로 정의된 경우)
- Dates는 항상 오브젝트입니다.
- Maths는 항상 오브젝트입니다.
- Regular Expressions(정규 표현식)는 항상 오브젝트입니다.
- Arrays는 항상 오브젝트입니다.
- Functions는 항상 오브젝트입니다.
- 오브젝트는 항상 오브젝트입니다.

원시 값(primitive values, 프리미티브)을 제외한 모든 JavaScript 값은 오브젝트입니다.

JavaScript Primitives

원시 값(primitive value)은 속성 또는 메서드가 없는 값입니다.

원시 데이터 타입(primitive data type)은 원시 값이 있는 데이터입니다.

JavaScript는 5 가지 유형의 원시 데이터 타입을 정의합니다.

- `string`
- `number`
- `boolean`
- `null`
- `undefined`

원시 값은 변경할 수 없습니다 (하드 코딩되어 변경될 수 없음)

x = 3.14이면 x 값을 변경할 수 있습니다. 그러나 3.14의 값은 변경할 수 없습니다.

value	**Type**	**Comment**
“Hello”	string	“Hello” is always “Hello”
3.14	number	3.14 is always 3.14
true	boolean	true is always true
false	boolean	false is always false
null	null (object)	null is always null
undefined	undefined	undefined is always undefined

Objects are Variables

JavaScript 변수는 하나의 값을 포함할 수 있습니다.

```
// Create and display a variable:
let person = "John Doe";
document.getElementById("demo").innerHTML = person;
```

오브젝트도 변수입니다. 그러나 오브젝트는 많은 값을 포함할 수 있습니다.

값은 **name : value** 쌍 (콜론으로 구분된 이름과 값)으로 작성합니다.

```
let person = {
  firstName: "John",
  lastName: "Doe",
  age: 50,
  eyeColor: "blue"
};

document.getElementById("demo").innerHTML = person.firstName + " " +
person.lastName;  // John Doe
```

```
JavaScript 객체는 명명된 값의 모음입니다.
```

Object Properties

JavaScript 오브젝트에서 명명된 값을 **속성(properties)**이라고 합니다.

Property	Value
firstName	John
lastName	Doe
age	50
eyeColor	blue

이름 값 쌍(name value pairs)으로 작성된 오브젝트는 다음과 유사합니다:

- PHP의 연관 배열(Associative arrays)
- Python의 사전(Dictionaries)
- C의 해시 테이블(Hash tables)
- Java의 해시 맵(Hash maps)
- Ruby 및 Perl의 해시(Hashes)

Object Methods

메서드는 오브젝트에 대해 수행할 수 있는 **작업(actions)**입니다.

원시 값, 다른 오브젝트 및 함수는 오브젝트 속성이 될 수 있습니다.

오브젝트 메서드는 **함수 정의(function definition)**를 포함하는 개체 속성입니다.

Property	Value
firstName	John
lastName	Doe
age	50
eyeColor	blue
fullName	function() {return this.firstName + “ ” + thisllastName;}

JavaScript는 속성(properties)과 메서드(methods)라고 불리는 명명된 값(named values)의 컨테이너입니다.

Creating a JavaScript Object

JavaScript를 사용하면 자신 만의 오브젝트를 정의하고 만들 수 있습니다.

새로운 오브젝트를 만드는 여러 가지의 방법들이 있습니다.

- 오브젝트 리터럴을 사용하여 단일 오브젝트를 정의하고 만듭니다.
- new 키워드를 사용하여 단일 오브젝트를 정의하고 만듭니다.
- 오브젝트 생성자를 정의한 다음, 생성된 타입의 오브젝트를 만듭니다.

```
ECMAScript 5에서는 ''Object.create()'' 함수를 사용하여 객체를 만들 수도 있습니다.
```

Using an Object Literal

이것은 JavaScript 오브젝트를 만드는 가장 쉬운 방법입니다.

오브젝트 리터럴을 사용하면 하나의 스테이트먼트에서 오브젝트를 정의하고 만들 수 있습니다.

오브젝트 리터럴은 중괄호 {} 안에 있는 이름:값 쌍 (예, age:50)의 목록입니다.

다음 예제는 네 가지 속성을 가진 새로운 JavaScript 오브젝트를 생성합니다.

```
let person = { firstName: "John", lastName: "Doe", age: 50, eyeColor: "blue"
};

document.getElementById("demo").innerHTML = person.firstName + " is " +
person.age + " years old.";
```

공백(spaces)과 줄 바꿈(line breaks)은 중요하지 않습니다. 오브젝트 정의는 여러 줄에 걸쳐있을 수 있습니다.

```
let person = {
  firstName: "John",
  lastName: "Doe",
  age: 50,
  eyeColor: "blue"
};

document.getElementById("demo").innerHTML = person.firstName + " is " +
person.age + " years old.";  // John is 50 years old.
```

Using the JavaScript Keyword new

다음의 예제도 4개의 속성(properties)을 가진 새로운 JavaScript 오브젝트를 생성합니다.

```
let person = new Object();
person.firstName = "John";
person.lastName = "Doe";
person.age = 50;
person.eyeColor = "blue";

document.getElementById("demo").innerHTML = person.firstName + " is " +
person.age + " years old.";  // John is 50 years old.
```

위의 두 예제는 정확히 동일합니다. ''new Object()''를 사용할 필요가 없습니다.
단순성(simplicity), 가독성(readability) 및 실행 속도(execution speed)를 위해서는, 첫 번째 방법 (오브젝트트 리터럴 방법)을 사용하십시오.

JavaScript Objects are Mutable

오브젝트는 변경할 수 있습니다: 오브젝트는 값(value)이 아닌 참조(reference)로 처리됩니다.

person이 오브젝트인 경우, 다음의 스테이트먼트는 person의 사본을 생성하지 않습니다.

```
var x = person; // This will not create a copy of person.
```

오브젝트 x는 person의 복사본이 아닙니다. person입니다. x와 person은 모두 같은 오브젝트입니다.

x를 변경하면 person도 변경됩니다. x와 person은 같은 오브젝트이기 때문입니다.

```
let person = { firstName: "John", lastName: "Doe", age: 50, eyeColor: "blue"
}

let x = person;
x.age = 10;

document.getElementById("demo").innerHTML = person.firstName + " is " +
person.age + " years old.";  // John is 10 years old.

console.log(person); // {firstName: "John", lastName: "Doe", age: 10,
eyeColor: "blue"}
console.log(x);      // {firstName: "John", lastName: "Doe", age: 10,
eyeColor: "blue"}
```

오션, Javascript, Objects, Definitions

From:
http://125.132.25.164/dokuwiki/ - **문제를 잘 정의하는 것은 문제를 절반 해결한 것이다. - 2023.12**

Permanent link:
http://125.132.25.164/dokuwiki/doku.php?id=wiki:javascript:javascript_note:js_object_definitions&rev=1619667956

Last update: **2022/03/10 19:52**