Javascript Objects

- description : Javascript Object Constructors
- author : 오션
- email : shlim@repia.com
- lastupdate : 2021-05-03

The source of this article

Javascript Object Constructors

```
// Constructor function for Person Objects
function Person(first, last, age, eye) {
  this.firstName = first;
  this.lastName = last;
  this.age = age;
  this.eyeColor = eye;
}

// Create a Person object
let myFather = new Person("Anthony", "Young", 50, "green");

// Display age
document.getElementById("demo").innerHTML =
"My father is " + myFather.age + ".";      // My father is 50.
```

대문자 첫 번째 철자로 생성자 함수(constructor function)의 이름을 지정하는 것이 좋습니다.

Object Types (Blueprints) (Classes)

이전 챕터(JS Object Constructors)의 예제들은 제한적입니다. 그 예제들은 하나의 오브젝트만 만듭니다.

때때로 우리는 동일한 “type”의 많은 오브젝트를 생성하기 위해 “ 청사진 ” 이 필요합니다.

“ 오브젝트 타입 ” 을 만드는 방법은 **오브젝트 생성자 함수(object constructor function)**를 사용하는 것입니다.

위의 예제에서 function Person()은 오브젝트 생성자 함수입니다.

new 키워드를 사용하여 생성자 함수를 호출하여 동일한 타입의 오브젝트를 만듭니다.

```
// Constructor function for Person objects
function Person(first, last, age, eye) {
  this.firstName = first;
  this.lastName = last;
  this.age = age;
  this.eyeColor = eye;
}

// Create two person objects
let myFather = new Person("John", "Doe", 50, "blue");
let myMother = new Person("Sally", "Rally", 48, "green");

// Display age
document.getElementById("demo").innerHTML =
  "My father is " + myFather.age + ". My mother is " + myMother.age + ".";
  // My father is 50. My mother is 48.
```

The this Keyword

JavaScript에서 this를 호출하는 것은 코드를 “ 소유 ” 하는 오브젝트입니다.

this의 값은 오브젝트에서 사용될 때 오브젝트 자체입니다.

생성자 함수에서, this는 값이 없고, 새로운 오브젝트를 대체합니다.
this의 값은 새 오브젝트가 생성될 때 새 오브젝트가 됩니다.

''this''는 변수가 아닙니다. 키워드입니다. ''this''의 값은 변경할 수 없습니다.

Adding a Property to an Object

기존의 오브젝트에 새로운 프로퍼티를 추가하는 것은 쉽습니다.

Example

```
// Constructor function for Person objects
function Person(first, last, age, eye) {
  this.firstName = first;
  this.lastName = last;
```

```
  this.age = age;
  this.eyeColor = eye;
}

// Create 2 Person objects
let myFather = new Person("John", "Doe", 50, "blue");
let myMother = new Person("Sally", "Rally", 48, "green");

// Add nationality to first object
myFather.nationality = "English";

// Display nationality
document.getElementById("demo").innerHTML =
  "My father is " + myFather.nationality;   // My father is English
```

프로퍼티는 myMother가 아닌(다른 person 오브젝트가 아닌) myFather에 추가됩니다.

Adding a Method to an Object

기존의 오브젝트에 새로운 메서드를 추가하는 것은 쉽습니다:

Example

```
// Constructor function for Person Objects
function Person(first, last, age, eye) {
  this.firstName = first;
  this.lastName = last;
  this.age = age;
  this.eyeColor = eye;
}

// Create 2 Person objects
let myFather = new Person("John", "Doe", 50, "blue");
let myMother = new Person("Sally", "Field", 48, "green");

// Add a name method to first object
myFather.name = function () {
  return this.firstName + " " + this.lastName;
};

// Display full Name
document.getElementById("demo").innerHTML =
  "My father is " + myFather.name();    // My father is John Doe
```

Adding a Property to a Constructor

기존의 오브젝트에 새로운 프로퍼티를 추가한 방식으로 오브젝트 생성자에 새로운 프로퍼티를 추가할 수 없습니다.

Example

```
// Constructor function for Person objects
function Person(first, last, age, eye) {
  this.firstName = first;
  this.lastName = last;
  this.age = age;
  this.eyeColor = eye;
}

// You can NOT add a new property to a constructor function
Person.nationality = "English";

// Create 2 Person objects
let myFather = new Person("John", "Doe", 50, "blue");
let myMother = new Person("Sally", "Rally", 48, "green");

// Display nationality
document.getElementById("demo").innerHTML =
  "The nationality of my Father is " + myFather.nationality;
    // The nationality of my Father is undefined
```

생성자에 새로운 프로퍼티를 추가하기 위해, 생성자 함수에 새로운 프로퍼티를 추가해야 합니다.

Example

```
// Constructor function for Person objects
function Person(first, last, age, eye) {
  this.firstName = first;
  this.lastName = last;
  this.age = age;
  this.eyeColor = eye;
  this.nationality = "English";
}

// Create 2 Person objects
let myFather = new Person("John", "Doe", 50, "blue");
let myMother = new Person("Sally", "Rally", 48, "green");

// Display nationality
```

```
document.getElementById("demo").innerHTML =
  "My father is " + myFather.nationality + ". My mother is " +
myMother.nationality;
   // My father is English. My mother is English
```

위와 같은 방식으로 오브젝트 프로퍼티는 기본 값(default values)를 가질 수 있습니다.

Adding a Method to a Constructor

생성자 함수는 또한 메서드를 정의할 수 있습니다.

Example

```
// Constructor function for Person Objects
function Person(first, last, age, eye) {
  this.firstName = first;
  this.lastName = last;
  this.age = age;
  this.eyeColor = eye;
  this.name = function () {
    return this.firstName + " " + this.lastName
  };
};

// Create a Person object
let myFather = new Person("John", "Doe", 50, "blue");

// Display full name
document.getElementById("demo").innerHTML =
"My father is " + myFather.name();  // My father is John Doe
```

기존의 오브젝트에 새로운 메서드를 추가한 방식과 동일하게 오브젝트 생성자에 새로운 메서드를 추가할 수 없습니다.

오브젝트 생성자에 메서드를 추가하는 것은 생성자 함수 내부에서 이루어져야 합니다.

Example

```

```

changeName() 함수는 person의 lastName 프로퍼티에 이름 값을 할당합니다.

Now You Can Try:

JavaScript는 **this**를 myMother로 " 대체 " 하여 언급하는person을 알고 있습니다.

Built-in JavaScript Constructors

JavaScript에는 네이티브 오브젝트에 대한 내장 생성자(built-in constructor)가 있습니다.

Example

Did You Know?

위에서 볼 수 있듯이, JavaScript에는 원시 데이터 타입(primitive data types)인
`String`, `Number` 및 `Boolean`의 오브젝트 버전이 있습니다.
그러나 복잡한 오브젝트를 만들 이유가 없습니다. 원시 값(primitive values)이 훨씬 빠릅니다.

또한:

`new Object()` 대신 오브젝트 리터럴 `{}`을 사용합니다.

`new String()` 대신 문자열 리터럴 `""`을 사용하십시오.

`new Number()` 대신 숫자 리터럴 `12345`를 사용하십시오.

`new Boolean()` 대신 불리언 리터럴 `true / false`를 사용하십시오.

`new Array()` 대신 배열 리터럴 `[]`을 사용하십시오.

`new RegExp()` 대신 패턴 리터럴 `/()/`을 사용하십시오.

`new Function()` 대신 함수 표현식 `() {}`을 사용하십시오.

Example

오션, Javascript, Objects, Constructors

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_constructors&rev=1620018229

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