# JavaScipt Naming Conventions

- description : JavaScipt Naming Conventions
- author : 오션
- email : shlim@repia.com
- lastupdate : 2022-04-06 Wed

## The source of this article

JavaScipt Naming Conventions

# 10 JavaScript Naming Conventions Every Developer Should Know

표준 명명 규칙을 따르면 가독성이 높아지고 코드를 더 쉽게 이해할 수 있습니다. 그러나 많은 개발자가 명명 규칙을 올바르게 사용하는 방법을 모르고 때로는 상황을 복잡하게 만듭니다.

이 글에서는 따라야 할 10가지 JavaScript 명명 규칙 모범 사례에 대해 설명합니다.

## 1. Naming Convention for Variables

JavaScript 변수 이름은 대소문자를 구분합니다. 소문자와 대문자는 구별됩니다. 예를 들어 다음과 같이 개 이름을 저장하기 위해 세 개의 고유 변수를 정의할 수 있습니다.

```
var DogName = 'Scooby-Doo';
var dogName = 'Droopy';
var DOGNAME = 'Odie';
console.log(DogName); // "Scooby-Doo"
console.log(dogName); // "Droopy"
console.log(DOGNAME); // "Odie"
```

그러나 JavaScript 변수를 선언하는 가장 권장되는 방법은 **camel case** 변수 이름을 사용하는 것입니다. JavaScript의 모든 유형의 변수에 대해 camel case 명명 규칙을 사용할 수 있으며, 동일한 이름을 가진 변수가 여러 개 존재하지 않도록 합니다.

```
// bad
var dogname = 'Droopy';
```

```
// bad
var dog_name = 'Droopy';
// bad
var DOGNAME = ‘Droopy’;
// bad
var DOG_NAME = 'Droopy';
// good
var dogName = 'Droopy';
```

변수 이름은 자명(자체를 잘 설명)해야 하며 저장된 값을 서술해야 합니다. 예를 들어 개의 이름을 저장할 변수가 필요한 경우, 그저 Name 대신에 더 의미가 있는 dogName을 사용해야 합니다.

```
// bad
var d = 'Scooby-Doo';
// bad
var name = 'Scooby-Doo';
// good
var dogName = 'Scooby-Doo';
```

## 2. Naming Convention for Booleans

Boolean 변수에 관해서는, 접두사로 is 또는 has를 사용해야 합니다. 예를 들어 강아지에게 주인이 있는지 확인하기 위해 Boolean 변수가 필요한 경우, 변수 이름으로 hasOwner를 사용해야 합니다.

```
// bad
var bark = false;
// good
var isBark = false;
// bad
var ideal = true;
// good
var areIdeal = true;
// bad
var owner = true;
// good
var hasOwner = true;
```

## 3. Naming Convention for Functions

JavaScript 함수 이름도 대소문자를 구분합니다. 따라서 변수와 유사하게 camel case 접근 방식은 함수 이름을 선언하는 데 권장되는 방법입니다.

그 외에도, 설명하는 명사와 동사를 접두사로 사용해야 합니다. 예를 들어 이름을 가져오는 함수를 선언하는 경우, 함수 이름은 getName이어야 합니다.

```
// bad
function name(dogName, ownerName) {
  return '${dogName} ${ownerName}';
}

// good
function getName(dogName, ownerName) {
  return '${dogName} ${ownerName}';
}
```

## 4. Naming Convention for Constants

JavaScript 상수도 대소문자를 구분합니다. 그러나 이러한 상수는 변경되지 않는 변수이므로 대문자로 작성해야 합니다.

```
var LEG = 4;
var TAIL = 1;
var MOVABLE = LEG + TAIL;
```

변수 선언 이름에 둘 이상의 단어가 포함된 경우, **UPPER_SNAKE_CASE**를 사용해야 합니다.

```
var DAYS_UNTIL_TOMORROW = 1;
```

모든 상수는 파일, 메서드 또는 클래스의 시작 부분에 정의되어야 합니다.

## 5. Naming Convention for Classes

JavaScript 클래스의 명명 규칙은 함수와 매우 유사합니다. 클래스의 기능을 설명하는 서술형 제목을 사용해야 합니다.

함수와 클래스 이름의 주요 차이점은 클래스 이름에 **Pascal case**를 사용해야 한다는 것입니다.

```
class DogCartoon {
  constructor(dogName, ownerName) {
    this.dogName = dogName;
    this.ownerName = ownerName;
  }
}

var cartoon = new DogCartoon('Scooby-Doo', 'Shaggy');
```

## 6. Naming Convention for Components

JavaScript 컴포넌트는 React와 같은 프론트엔드 프레임워크에서 널리 사용됩니다. 컴포넌트는 DOM에서 사용되지만, 컴포넌트를 클래스와 유사하게 처리하고 Pascal case를 사용하여 이름을 정의하는 것을 권장합니다.

```
// bad
function dogCartoon(roles) {
  return (
    < div >
      < span > Dog Name: { roles.dogName } < /span>
      < span > Owner Name: { roles.ownerName } < /span>
    < /div>
  );
}

// good
function DogCartoon(roles) {
  return (
    < div >
      < span > Dog Name: { roles.dogName } < /span>
      < span > Owner Name: { roles.ownerName } < /span>
    < /div>
  );
}
```

첫 글자는 항상 대문자로 작성되기 때문에, 컴포넌트를 활용하면 네이티브 HTML 및 웹 컴포넌트와 차별화됩니다.

```
<div>
  <DogCartoon
    roles={{ dogName: 'Scooby-Doo', ownerName: 'Shaggy' }}
  />
</div>
```

## 7. Naming Convention for Methods

몇몇 차이점이 있지만, JavaScript 함수와 메서드의 구조는 매우 유사합니다. 따라서 명명 규칙 규칙은 동일합니다.

우리는 JavaScript 메소드를 선언하기 위해 ㅊ**camel case**를 사용해야 하고 이름을 더 의미 있게 만들기 위해 동사를 접두사로 사용해야 합니다.

```
class DogCartoon {
  constructor(dogName, ownerName) {
    this.dogName = dogName;
    this.ownerName = ownerName;
  }

  getName() {
    return '${this.dogName} ${this.ownerName}';
  }
}

var cartoon= new DogCartoon('Scooby-Doo', 'Shaggy');

console.log(cartoon.getName());
// "Scooby-Doo Shaggy"
\\

===== 8. Naming Convention for Denoting Private Functions =====
밑줄( _ )은 MySQL 및 PHP와 같은 언어에서 변수, 함수 및 메서드를 정의하는 데 널리 사용됩니다.
그러나 JavaScript에서 밑줄은 private 변수나 함수를 나타내는 데 사용됩니다.\\
\\
예를 들어 toonName과 같은 프라이빗 함수 이름이 있는 경우 접두사(_toonName)로 밑줄을 추가하여 프라이빗 함수로 표시할 수 있습니다.\\
\\
<code javascript>
class DogCartoon {
  constructor(dogName, ownerName) {
    this.dogName = dogName;
    this.ownerName = ownerName;
    this.name = _toonName(dogName, ownerName);
  }
  _toonName(dogName, ownerName) {
    return `${dogName} ${ownerName}`;
  }
}

var cartoon = new DodCartoon('Scooby-Doo', 'Shaggy');

// good
var name = cartoon.name;
console.log(name);
```

```
// "Scooby-Doo Shaggy"

// bad
name =cartoon._toonName(cartoon.dogName, cartoon.ownerName);
console.log(name);
// "Scooby-Doo Shaggy"
```

## 9. Naming Convention for Global Varialbes

</code>

오션,, JavaScipt, Naming, Conventions