JavaScript Const

- description : JavaScript Let
- author : 오션
- email : shlim@repia.com
- lastupdate : 2021-04-13

ECMAScript 2015

ES2015는 `let`과 `const`라는 두 가지 중요한 새 JavaScript 키워드를 도입했습니다.

재할당 할 수 없다는 점을 제외하고, `const`로 정의된 변수는 `let` 변수처럼 작동합니다.

```
<!DOCTYPE html>
<html lang="en">
<body>
    <h2>JavaScipt const</h2>
    <p>You cannot change a primitive value.</p>
    <p id="demo"></p>
    <script>
        try {
            const PI = 3.141592653589793;
            PI = 3.14;
        }
        catch (err) {
            document.getElementById("demo").innerHTML = err;
        }
    </script>
</body>
</html>
```

Block Scope

`const`로 변수를 선언하는 것은 **블럭 범위(Block Scope)**와 관련하여 `let`과 비슷합니다.

아래 예제의 블록 내부에서 선언된 x는 블록 외부에서 선언된 x와 동일하지 않습니다.

```
<!DOCTYPE html>
<html lang="en">
<body>
  <h2>Declaring a Variable Using const</h2>
  <p id="demo"></p>
```

```
  <script>
    var x = 10;
    // Here x is 10
    {
      const x = 2;
      // Here x is 2
    }
    // Here x is 10
    document.getElementById("demo").innerHTML = x;
  </script>
</body>
</html>
```

Assigned when Declared

JavaScript const 변수는 선언될 때 값을 할당받아야 합니다:

Incorrect

```
const PI;
PI = 3.14159265359;
```

Correct

```
const PI = 3.14159265359;
```

Not Real Constants

const 키워드는 약간 오해의 소지가 있습니다.

상수 값을 정의하지 않습니다. 값에 대한 상수 참조를 정의합니다.

이 때문에, 상수 프리미티브(primitive) 값은 변경할 수 없지만, 상수 오브젝트의 속성은 변경할 수 있습니다.

Primitive Values

상수(constant)에 프리미티브 값을 할당하면, 프리미티브 값을 변경할 수 없습니다.

```
const PI = 3.141592653589793;
PI = 3.14;      // This will give an error
```

```
PI = PI + 10;   // This will also give an error
```

Constant Object can Change

상수 오브젝트의 속성을 변경할 수 있습니다.

```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript const</h2>
  <p>Declaring a constant object does NOT make the objects properties
unchangeable:</p>
  <p id="demo"></p>
  <script>
    // Create an object:
    const car = { type: "Flat", model: "500", color: "white" };
    // Change a property:
    car.color = "red";
    // Add a property:
    car.owner = "Johnson";
    // Display the property:
    document.getElementById("demo").innerHTML = "Car owner is " + car.owner;
  </script>
</body>
</html>
```

하지만 상수 오브젝트를 재할당할 수는 없습니다.

```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript const</h2>
  <p>But you can NOT reassign a constant object:</p>
  <p id="demo"></p>
  <script>
    try {
      const car = { type: "Fiat", model: "500", color: "white" };
      car = { type: "Volvo", model: "EX60", color: "red" };   /* Error */
    }
    catch (err) {
      document.getElementById("demo").innerHTML = err;
    }
  </script>
</body>
</html>
```

Constant Arrays can change

상수 배열의 요소들을 변경할 수 있습니다.

```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript const</h2>
  <p>Declaring a constant array does NOT make the elements unchangeable:</p>
  <p id="demo"></p>
  <script>
    // Create an Array:
    const cars = ["Saab", "Volvo", "BMW"];
    // Change an element:
    cars[0] = "Toyota";
    // Add an element:
    cars.push("Audi");
    // Display the Arrays:
    document.getElementById("demo").innerHTML = cars;
  </script>
</body>
</html>
```

하지만, 상수 배열을 재할당할 수는 없습니다.

```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript const</h2>
  <p>You cna NOT reassign a constant array:</p>
  <p id="demo"></p>
  <script>
    try {
      const cars = ["Saab", "Volvo", "BMW"];
      cars = ["Toyota", "volvo", "Audi"];   /* Error */
    }
    catch (err) {
      document.getElementById("demo").innerHTML = err;
    }
  </script>
</body>
</html>
```

const 키워드는 Internet Explorer 10 이하 버전에서는 지원되지 않습니다.

Redeclaring

JavaScript var 변수를 재선언하는 것은 프로그램의 모든 위치에서 허용됩니다.

```
var x = 2;     // Allowed
var x = 3;     // Allowed
x = 4;         // Allowed
```

동일한 범위 또는 동일한 블록에서, 기존 var 또는 let 변수를 const에 재선언 또는 재할당하는 것은 허용되지 않습니다.

```
var x = 2;       // Allowed
const x = 2;     // Not Allowed
{
  let x = 2;     // Allowed
  const x = 2;   // Not Allowed
}
```

동일한 범위 또는 동일한 블록에서, 기존 const 변수를 재선언 또는 재할당하는 것은 허용되지 않습니다.

```
const x = 2;     // Allowed
const x = 3;     // Not Allowed
x = 3;           // Not Allowed
var x = 3;       // Not Allowed
let x = 3;       // Not Allowed

{
  const x = 2;   // Allowed
  const x = 3;   // Not Allowed
  x = 3;         // Not Allowed
  var x = 3;     // Not Allowed
  let x = 3;     // Not Allowed
}
```

다른 범위 또는 다른 블록에서 const로 변수를 재선언하는 것은 허용됩니다.

```
const x = 2;     // Allowed

{
  const x = 3;   // Allowed
}

{
  const x = 4;   // Allowed
}
```

Hoisting

var로 정의된 변수는 맨 위로 올라 와서 언제든지 초기화 될 수 있습니다.

의미 : 변수를 선언하기 전에 사용할 수 있습니다.

```
carName = "Volvo";
alert(carName);
var carName;
```

const로 정의된 변수는 블록의 맨 위에 올려지지만, 초기화되지는 않습니다.

의미 : 코드 블록은 변수를 인식하지만 선언될 때까지 사용할 수 없습니다.

선언될 때까지 변수는 블록이 시작되는 “TDZ (Temporal Dead Zone)“에 있습니다.

선언하기 전에 const 변수를 사용하는 것은 구문 오류이므로, 코드가 실행되지 않습니다.

This code will not run

```
carName = "Volvo";
const carName;
```

오션,, Javascript, Const

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_const&rev=1618312229

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