



호이스팅 (Hoisting)

목차

[호이스팅? \(hoisting\)](#)

[호이스팅 발생 과정](#)

[실행컨텍스트\(execution context\)](#)

호이스팅? (hoisting)

- hoisting) N. 끌어 올리기, 들어올려 나르기
- hoisting = move declaration from bottom to top
- 호이스팅은 변수 선언이 스코프[코드실행환경을 참조할 수 있는 유효범주]의 선두로 끌어 올려진 것처럼 동작하는 자바스크립트의 고유의 특징.
 - 함수, 변수가 실행되기 전에 안에있는 변수들을 범위의 최 상단으로 끌어올리는 것!
 - 일반적으로 변수 선언 후, 결과 값 확인 가능!

```
JS hoisting.js > ...
1  var a = 1
2  console.log(a)
```

문제 8 출력 디버그 콘솔 터미널 Code

[Running] node "/Users/mochiseung/elize6_practice/eg.js/hoisting.js"

1

하지만! 호이스팅은 변수 a가 선언되기 전에 a를 선언해도 스코프의 선두측, 변수가 유효할 수 있는 범위의 최 상단으로 끌어 올려진 것처럼 동작함!

- 변수 선언과, 초기화를 동시에!

따라서 “a를 찾을 수 없음”(ReferenceError: a is not defined) 의 에러가 아니라 undefined(선언은 되었지만 아무값도 지정되어 있지 않음) 을 확인할 수 있게 됨!

```
JS hoisting.js > ...
1 console.log(a)
2 var a = 1
3 console.log(a)

문제 8 출력 디버그 콘솔 터미널 Code
[Running] node "/Users/mochiseung/elize6_practice/eg.js/hoisting.js"
undefined
1
```

자바스크립트 데이터에 var a 라는 변수는 저장되어 있기 때문에 console.log(a)는 undefined라는 값을 나타내게 됨! 이것이 호이스팅

- 호이스팅은 실행 컨텍스트 과정에서 발생하게 된다!
- 참고) 2줄에서 변수에 값이 할당되고 그 값이 3번에서 출력됨.

호이스팅 발생 과정

▼ 실행컨텍스트(execution context)

자바스크립트의 동작 원리를 담고 있는 개념!

소스코드를 평가하는 과정에서 실행 컨텍스트를 생성하고 변수, 함수 등의 선언문만 먼저 실행하여 생성된 변수나 함수 식별자를 key로 실행 컨텍스트가 관리하는 스코프(렉시컬 환경의 환경 레코드)에 등록한다.

→ 따라서! 호이스팅이 발생하는 과정은 !

소스코드 평가 과정에서 변수 선언문 [var x;]가 실행되면 생성된 변수 식별자 x는 실행 컨텍스트가 관리하는 스코프에 등록되고 undefined로 초기화 되는 것이다.

- 참고) 함수에도 호이스팅이 있다!

```
JS hoisting.js > add
1 console.log (add(2,5));
2
3 function add(x,y){
4   return x+y;
5 }

문제 8 출력 디버그 콘솔 터미널 Code
[Running] node "/Users/mochiseung/elice6_practice/eg.js/hoisting.js"
7
```

함수 선언문이 코드를 선두로 끌어 올려진 것처럼 동작하는 것이 바로 함수 호이스팅!

코드가 한 줄씩 순차적으로 실행되기 시작하는 런타임에 이미 함수 객체가 생성되어 있고, 함수 이름과 동일한 식별자에 할당까지 완료된 상태로 함수 선언문 이전에 함수를 참조할 수 있으며 호출도 가능하다!

- 참고1) var 키워드를 사용한 변수 선언문과 함수 선언문은 런타임 이전에 자바스크립트 엔진에 의해 먼저 실행되어 식별자를 생성한다는 점에서 동일하다!
- 참고2) var 키워드로 선언된 변수는 undefined로 초기화 된다. 하지만 함수 선언문을 통해 암묵적으로 생성된 식별자는 함수 객체로 초기화되어 호출이 된다는 차이점이 있다.
- 참고3) 함수 표현식으로 함수를 정의할 경우 함수 호이스팅이 발생하는 것이 아니라 변수 호이스팅이 발생된다.

var 호이스팅 이대로 괜찮은가? : let의 등장!

- var는 변수가 중복 선언이 된다!

```
JS hoisting.js > ...
1  var a = 1
2  console.log(a)
3  var a = 2
4  console.log(a)
```

문제 8 출력 디버그 콘솔 터미널 Code

[Running] node "/Users/mochiseung/elice6_practice/eg.js/hoisting.js"

1
2

즉, 나와 같은 주민번호를 가진 사람이 2명이라는 뜻!

변수 선언문 이전에 변수를 참조하는 것은 변수 호이스팅에 에러를 발생시키지 않지만, 그 외에 중복선언 현상 및 무조건적인 호이스팅으로 프로그램의 흐름상 맞지 않는 결과를 창출 내며 가독성을 떨어뜨리고 오류를 발생시킬 여지를 남긴다.

그래서 등장한 것이 바로 let!

```
JS hoisting.js > ...
1  let a = 1
2  console.log(a)
3  let a = 2
4  console.log(a)
```

문제 10 출력 디버그 콘솔 터미널 Code

[Running] node "/Users/mochiseung/elice6_practice/eg.js/hoisting.js"

/Users/mochiseung/elice6_practice/eg.js/hoisting.js:3
let a = 2
 ^

SyntaxError: Identifier 'a' has already been declared
 at Object.compileFunction (node:vm:360:18)

var와 let의 비교!

```
JS hoisting.js > ...
1   for (var i = 1; i<5; i++){
2   |   console.log(i)
3   }
4   console.log(i)
```

문제 8 출력 디버그 콘솔 터미널 Code

[Running] node "/Users/mochiseung/elice6_practice/eg.js/hoisting.js"

1
2
3
4
5

var는 for문을 사용했을 때, 범위 밖에서도 호이스팅이 되어서 5가 출력이 된다. [var문은 함수만 지역변수로 호이스팅되고, 나머지는 모두 전역변수로 호이스팅이 된다는 특징을 가지고 있다.]

```
JS hoisting.js > ...
1   for (let i = 1; i<5; i++){
2   |   console.log(i)
3   }
4   console.log(i)
```

문제 8 출력 디버그 콘솔 터미널 Code

[Running] node "/Users/mochiseung/elice6_practice/eg.js/hoisting.js"

1
2
3
4
/Users/mochiseung/elice6_practice/eg.js/hoisting.js:4
console.log(i)
| |
| ^
ReferenceError: i is not defined

let의 경우에는 지역변수로 선언되어, 변수 외의 것은 영향을 주지 못하게 되는 모습을 보일 수 있다. 따라서 var의 호이스팅과, 중복선언 등에 대한 문제점 해결 완료!

참고) let도 호이스팅은 가능하지만, Temporal Death Zone으로 인해, 선언문이 나오기 전 까지 일시적으로 변수에 접근하지 못하게 함.

tip!


hoisting 개념을 이해하기 위해서는 스코프 / 실행컨텍스트로 큰 그림을 그리고, 공부하면 좋아요 =)

참고

도서

모던 자바스크립트 Deep Dive

자바스크립트를 둘러싼 기본 개념을 정확하고 구체적으로 설명하고, 자바스크립트 코드의 동작 원리를 집요하게 파헤친다. 작성한 코드가 컴퓨터 내부에서 어떻게 동작할 것인지 예측하고, 명확히 설명할 수 있

 https://www.aladin.co.kr/shop/wproduct.aspx?ISBN=K282633473&start=pnaver_02

Deep Dive

자바스크립트의 기본 개념과 동작 원리

프로그래밍 & 컴퓨터 시스템, 2016

이동호 지음



youtube

<https://www.youtube.com/watch?v=fETYLCU2YYc>

[코딩만화] Scope가 뭔가요? (feat: let, const, var의 차이)

 https://youtu.be/HsJ4oy_jBx0

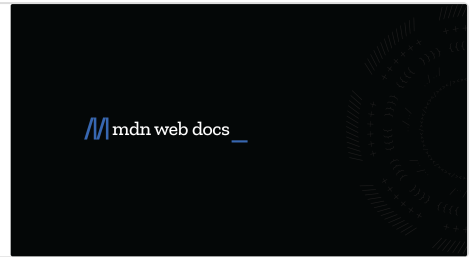


블로그


호이스팅 - MDN Web Docs 용어 사전: 웹 용어 정의 | MDN

JavaScript는 초기화를 제외한 선언만 호이스팅합니다. 변수를 먼저 사용하고 그 후에 선언 및 초기화가 나타나면, 사용하는 시점의 변수는 기본 초기화 상태(var 선언 시 undefined, 그 외에는 초기화하지 않

 <https://developer.mozilla.org/ko/docs/Glossary/Hoisting>



siempre hermosa : 네이버 블로그

 https://blog.naver.com/siempre_hermosa/222923891942

책,여행 그리고 IT Tech : 네이버 블로그

 <https://blog.naver.com/dltjddms126/222896623262>