4 장 변수

변수

변수는 기억장치 같은거로 생각하면 편함.

자바스크립트에서 메모리는 가비지 컬렉션을 이용해 자동으로 관리됨

cpu에서 계산된 결과값을 메모리에 저장하고, 그 저장된 값을 쉽게 관리하기 위해 변수를 사용함.

식별자

변수의 이름을 식별자라고 함.

쉽게 우리가 부를 이름이라고 보면될듯함.
변수 이름은 첫아이 이름을 짓듯 심사숙고해서 지어야 한다고 되어 있는데 , 개인적으론 그 정도까지 심사숙고해본적은 없음.

네이밍 규칙들이 있음.

변수의 선언

자바스크립트에 변수는 var만 존재했었음, 하지만 이젠 let이나 const가 생겼는데 이는 다른 챕터에서 알아보나봄.
그래도 다음 내용을 설명하기 쉽게 하기위해 설명하자면 let은 var랑 비슷한 일반 변수라고 생각하면되고, const는 상수로 알고있음.

```
var temp123;
```

값을 할당하진 않은 상태이고, 변수를 사용하겠다고 선언해둔 상태임.
위처럼 사용할 시 자바스크립트는 temp123이라는 식별자를 이름으로 하는 메모리 공간을 예약해두게 됨.

값을 지정해주지 않았기에 undefind 라는 값이 자동으로 할당되어 초기화됨.

var 키워드를 사용한 선언은 선언단계와 초기화 단계가 '동시에' 진행됨.
일반적으로 초기화란 변수가 선언된 이후 최초로 값을 할당하는 것을 말함.
그래서 var 키워르도 선언한 변수는 어떠한 값도 할당하지 않아도 undefined라는 값을 가

짐.
이 부분이 포인트인거 같으니 기억해두시길 바람.

그리고 변수는 아래와 같이 한번에 여러개를 선언할 수도 있음.
다중 할당도 가능한데, 이는 아래에서 알아보겠음.

```
var temp1, temp2, temp3, temp4, lego, somi;
```

다만, 가독성의 문제로 권장하지 않는다고 함.

undefined는 자바나 C++의 null이나 파이썬의 None과 비슷하면서도 조금 다름.

변수 호이스팅

자바스크립트의 코드는 인터프리터에 의해 위에서 부터 한줄씩 순차적으로 실행됨

즉, 일반적으론 이런 코드가 불가능함

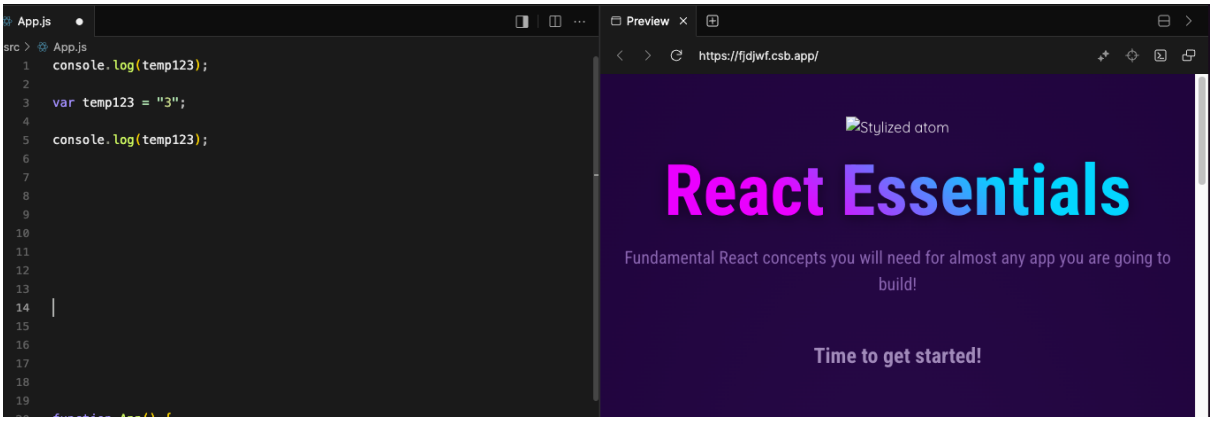
```
console.log(temp123);

var temp123 = "123";
```

하지만 자바스크립트에선 위의 코드가 가능함.

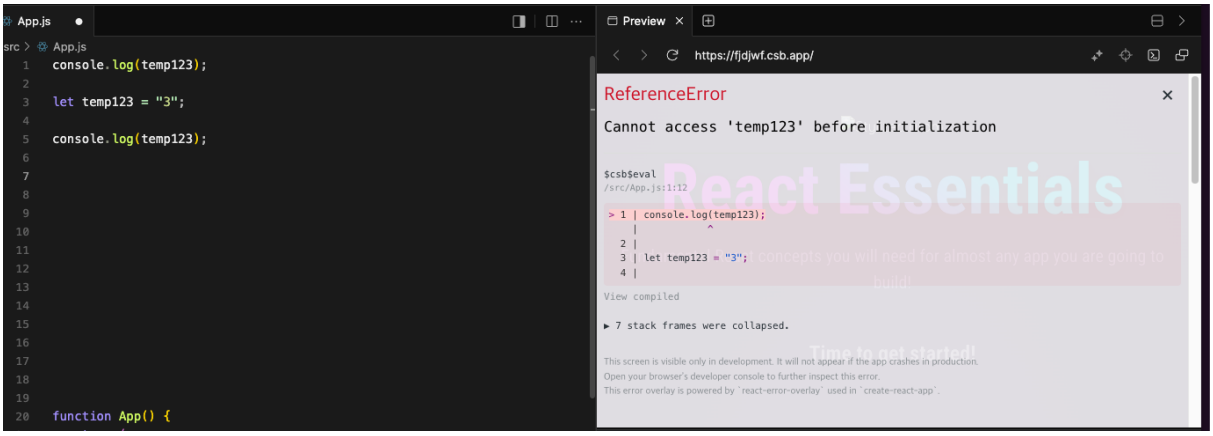
자바스크립트 엔진이 소스코드를 실행하기 전에 소스 코드의 평가과정을 거치면서 소스코드를 실행 할 준비를 하는데, 이때 자바스크립트 엔진이 변수를 선언을 포함한 모든 선언문(변수, 함수 선언문 등)을 찾아서 먼저 실행한 뒤 소스코드를 실행한다고 함.

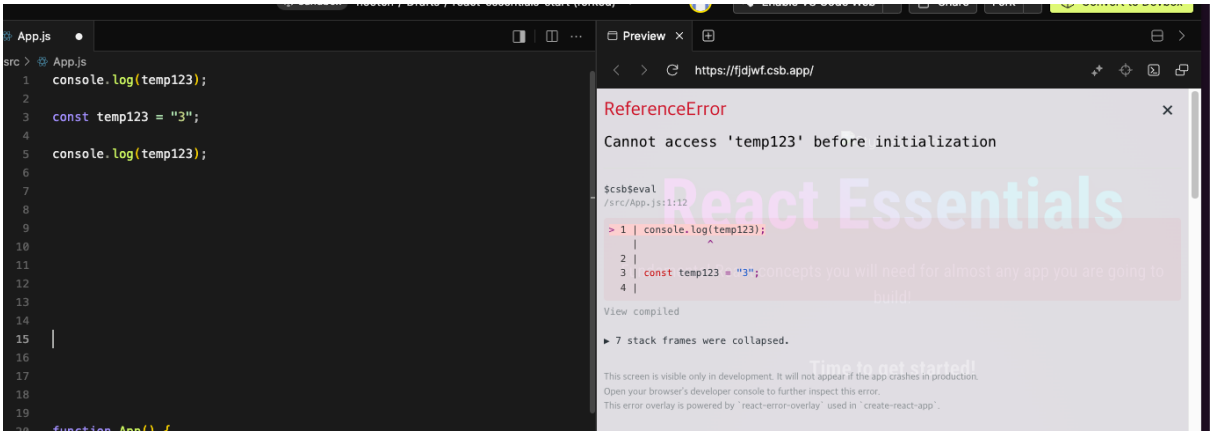
어디에 쓸 수 있을진 아직 잘 모르겠지만 신기해서 실험해봤는데, 근데 이게 또 책 설명이랑은 결과가 좀 다름.



일단 var경우엔 책에서 설명한 대로 잘 되었음.

그런데 let이나 const를 사용할땐 결과가 약간 달랐음.





인터넷 검색해보니 이게 일단 호이스팅 되고 있는거라고 함!

그래서 이해는 안가지만, 마치 이런케이스라고 생각하고 넘기려고 하였음.

그런데 막상 nodejs를 통해 실행시켜보아도 에러만 뜨고 호이스팅이 제대로 이루어지는지 알 도리가 없었음.

일단 이 문제는 살짝 뒤로 재치고, 갑작스럽지만 자바스크립트 변수 생성절차부터 설명하겠음.

자바스크립트의 변수 생성 단계

자바스크립트에서는 총 3단계에 걸쳐서 변수를 생성한다.

- **선언(Declaration)**: 스코프와 변수 객체가 생성되고, 스코프가 변수 객체를 참조한다. *초기화 전까지는 TDZ 상태이다.*
- **초기화(Initialization)**: 변수 객체 값을 위한 공간을 메모리에 할당한다. 이 때 할당되는 값은 undefined다. myValue = undefined
- **할당(Assignment)**: 변수 객체에 값을 할당한다. myValue = 150

출처(\[JavaScript\] 변수와 함수 호이스팅\(Hoisting\)에 대해 알아보자)

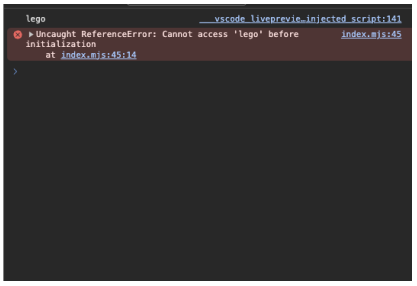
하지만 var는 아까 변수 선언에서 설명하였듯 선언과 동시에 초기화가 이루어짐.
그래서 이미 값으로 undefined가 할당되어 있기에 에러가 안떳던 것임.

하지만 let, const는 선언까지만 진행 될 뿐, 초기화가 이루어 지지 않아 메모리에 할당되지 않은 상태.
즉 TDZ 상태인거임. 그래서 에러가 나는 것.

그래도 에러가 나는데 호이스팅이 된다는 부분이 살짝 이해가 가지 않아서 조금 더 알아보기로 하였음.

우선 var의 경우임. 사실 위와 크게 다른건 없음. 하지만 호이스팅이 일어나는 시기를 좀더 자세히 알아 볼 수 있음.

아래는 let의 경우임.



let 또한 진짜로 호이스팅을 함.
그러므로 글로벌 스코프에서 이미 let lego = “lego”를 선언하여 값까지 할당하여 첫번째 log는 “lego”를 뱉어내었지만 ,밑의 블록 스코프 내에서의 두번째 let legos = “lego”; 는 호이스팅은 일어났지만 TDZ상태인 것이고, 바로 윗줄에 console.log(lego); 가 있어서 에러를 뱉어내게 된 것임.

변수값 할당

```
var temp1234;
temp123 = 123;
```

or

```
var temp123 = 123;
```

변수에 값은 위처럼 할당할 수 있음.

주의점으로는 변수의 선언과 값의 할당은 실행 시점이 다름.
이는 앞에 설명했던 소스 코드 실행 이전에 자바스크립트 엔진이 평과 과정을 거쳐서....하는 그 부분임.

예)

```
console.log(temp123);
//결과 콘솔값 undefined

var temp123 = 123;
```

```
console.log(temp123);
//결과 콘솔값 123
```

변수 값의 재할당

```
var temp123 = 123;
temp123 = 1234567890;
```

변수 값의 재할당은 위와 같이 이루어짐.
쉽게 temp123이 갖고 있는 값이 123이 아닌 1234567890으로 바뀐 것임.

하지만 내부에서 메모리는 123이 저장된 곳에 1234567890을 덮어씌운 것이 아니라, 123이 저장된 메모리를 지우고, 새로 메모리 주소를 확보하여 거기에 1234567890을 저장 한 후, temp123에 새 메모리 주소를 엮어둔것임.

이때 저장되어있던 이전에123을 할당한 메모리 공간이 바로 지워지는 것이 아니라 나중에 가비지 콜렉터가 주기적으로 이러한 고아 메모리들을 모아서 한번에 지워줌.

식별자 네이밍 규칙

식별자는 다음과 같은 네이밍 규칙을 준수하여야함.

- 식별자는 특수문자를 제외한 문자, 숫자, 언더스코어, 달러,기호를 포함할 수 있다.
- 단, 식별자는 특수문자를 제외한 문자, 언더스코어(_), 달러 기호로 시작해야한다. 숫자로 시작하는 것은 허용하지 않는다.
- 예약어는 식별자로 사용할 수 없다.

예약어는 이미 자바스크립트에서 사용되고 있거나 사용 예정인 단어들을 말함.

예약어는 프로그래밍 언어에서 사용되고 있거나 사용될 예정인 단어를 말한다. 자바스크립트의 예약어는 다음과 같다.

await	break	case	catch	class	const
continue	debugger	default	delete	do	else
enum	export	extends	false	finally	for
function	if	implements*	import	in	instanceof
interface*	let*	new	null	package*	private*
protected*	public*	return	super	static*	switch
this	throw	true	try	typeof	var
void	while	with	yield*		

* 식별자로 사용 가능하나 strict mode에서는 사용 불가

표 4-1 예약어

식별자로 유니코드 사용 가능

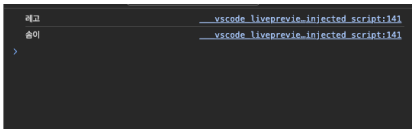
식별자로 유니코드 문자를 허용함.
그래서 한글 변수나 일본어변수 등을 사용할 수있으나 권장하지는 않는다고 함

```
var 레고 = "레고";
```

다중 선언 및 다중 할당

다중 선언이 되길래 할당도 있을 것 같아서 알아보았음.

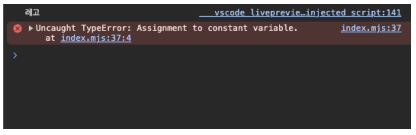
var의 경우



Var의 경우 스크린샷엔 나와있지 않지만, 변수 선언 이전에 호출을 작성 할 시, undefined 로 나오게됨.

---

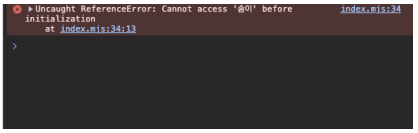
const의 경우



뒤의 솜이도 const가 적용되었는지 알아보기 위해 솜이를 레고로 바꾸어보았음.
에러내용과 같이 const가 적용되어 값 변경이 불가능함을 알 수 있음.

---

let의 경우



뒤의 솜이가 let이 적용되었는지 알아보기 위해 솜이를 호이스팅 해보았음.
보시다싶이 let은 호이스팅시 참조에러를 뿜어내므로 let 할당에 성공하였음을 알수있었음.

변수 이름

변수이름은 변수의 존재목적을 쉽게 이해할 수 있도록 의미를 명확히 표현해야 한다. 좋은 변수 이름은 코드의 가독성을 높인다.

네이밍 컨벤션

네이밍 컨벤션은 크게
네종류가 있음.

카멜 케이스

```
var puffTheMagicDragon
```

스네이크 케이스

```
var puff_the_magic_dragon
```

파스칼 케이스

```
var PuffTheMagicDragon
```

헝가리안 케이스는 안써본거라 좀 찾아봄
C언어 진영에서 주로 사용했었고, 최근엔 지양하고 있다고함. 방식은 변수명 앞에 자료형을 붙이는 식임.

```
var strLego = "lego";
var intAge = 5;
```

```
var floatWeight = 4.5;
```

일관성만 유지한다면 어떤 네이밍 컨벤션을 써도 관계는 없으나, 자바스크립트에서는 일반적으로 변수나 함수의 이름에는 카멜케이스를, 생성자 함수, 클래스 이름에는 파스칼 케이스를 사용한다고함.
