Closure in Javascript

#TIL/dev/language/javascript

1. Javascript는 함수를 리턴하게되면 함수 Pointer만 복사하는게 아니라, 함수의 lexical cope context까지 함께 만들어준다. 즉, 함수와 함수가 선언된 어휘적 환경의 조합을 클로져라고 한다!!

```
function makeAdder(x) {
  var y = 1;
  return function(z) {
    y = 100;
    return x + y + z;
  };
}

var add5 = makeAdder(5);
var add10 = makeAdder(10);
//클로저에 x와 y의 환경이 저장됨

console.log(add5(2));  // 107 (x:5 + y:100 + z:2)
console.log(add10(2)); // 112 (x:10 + y:100 + z:2)
//함수 실행 시 클로저에 저장된 x, y값에 접근하여 값을 계산
```

예컨대, 위처럼 makeAdder 내부의 함수는 리턴되면서 클로져를 만드는데, 그것은 makeAdder 내부의 lexical context와 함수 그자체로 이루어지며, 만들어질때마다 독립된 closure로 형성된다고 보면 되겠다.

클로저

주의점

자바스크립트를 사용할때 클로져를 정말 조심해야한다. 함수를 만듦으로 인한 개별 클로져 생성이 예상치못한 이슈를 유발하기도한다. react에서의 클로져이슈는 useCallback을 이용하면어떻게든 해결할 수 있다. 이해하지못할 상태가 발생한다면 클로져를 의심해보자.