Chap23 실행 컨텍스트

23-1~4

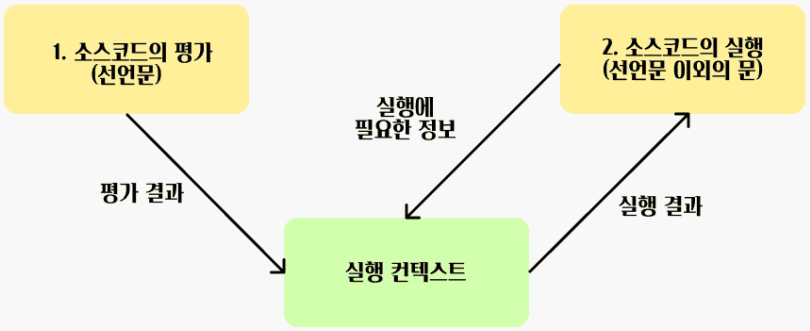
코드가 실행되려면?

1. 선언에 의해 생성된 모든 식별자(변수, 함수, 클래스 등)를 스코프를 구분하여 등록하고 상태변화(식별자에 바인딩된 값의 변화)를 지속적으로 관리할 수 있어야 한다.
2. 스코프는 중첩 관계에 의해 스코프 체인을 형성해야 한다. 즉 스코프 체인을 통해 상위 스코프로 이동하며 식별자를 검색할 수 있어야 한다.
3. 현재 실행 중인 코드의 실행 순서를 변경(예를 들어, 함수 호출에 의한 실행 순서 변경)할 수 있어야 하며 다시 되돌아 갈 수도 있어야 한다.

이러한 것들이 잘 관리되어야 한다. 이 관리하는 것이 바로 ***실행 컨텍스트***이다.

실행 컨텍스트는

- 소스코드를 실행하는데 필요한 환경을 제공하고 코드의 실행 결과를 실제로 관리하는 영역
- 식별자를 등록하고 관리하는 스코프아 코드 실행 순서 관리를 구현한 내부 메커니즘으로, 모든 코드는 실팽컨텍스트를 통해 실행되고 관리된다.
- LexicalEnvironment 컴포넌트와 VariableEnvironment 컴포넌트로 구성된다.

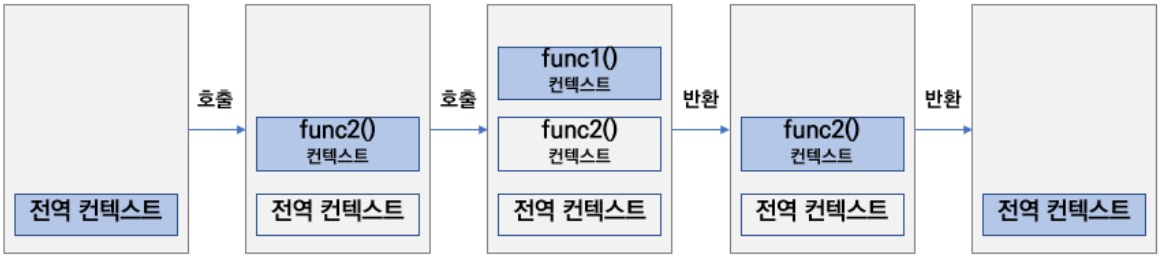


그리고 실행컨텍스트는 스택 자료구조로 관리를 한다. 이를 ***실행 컨텍스트 스택***이라고 부른다.

```
console.log("전역 컨텍스트입니다.");

function func1(){
    console.log("func1 입니다.");
}

function func2(){
    func1();
    console.log("func2 입니다.");
}
func2();
```



최상위에 존재하는 실행 컨텍스트는 언제나 현재 실행 중인 코드의 실행 컨텍스트다. 이를 ***실행 중인 실행 컨텍스트***라고 부른다.

23-5 렉시컬 환경

렉시컬 환경

- 식별자와 식별자에 바인딩된 값, 그리고 상위 스코프에 대한 참조를 기록하는 자료구조로 실행 컨텍스트를 구성하는 컴포넌트다.
- **실행 컨텍스트 스택**이 코드의 실행 순서를 관리한다면 **렉시컬 환경**은 스코프와 식별자를 관리한다.

23-7 실행 컨텍스트와 블록 레벨 스코프

- var키워드로 선언한 변수는 오로지 함수의 코드 블록만 지역 스코프로 인정하는 함수 레벨 스코프를 따른다.
- let,const 키워드로 선언한 변수는 모든 코드블록을 지역 스코프로 인정하는 블록레벨 스코프를 따른다.