9일간의 자바스크립트

동시성 프로그래밍

동시성 프로그래밍이란?

concurrent programming

```
function a() {
  for (let i = 0; i < 1000000; i++) {
    new Date().toString();
  }
}

function b() {
  console.log("please call me asap!!!!!!");
}

a();
b();
```

- 가정을 하나 해봅시다
- 시간이 오래 걸리는 작업 A가 있고, 빨리 끝내야 하는 작업 B가 있다고 하자.
- A를 먼저 실행을 했으면 B는 빨리 끝내야 함에도 불구하고 A가 끝나기 전 까지는 실행하지를 못한다.
- 하나의 라인을 타고 코드가 흘러간다고 보면 된다. (단일 쓰레드)

동시성 프로그래밍이란?

concurrent programming

- 그래서 동시에 다양한 처리를 행하는 동시성 프로그래밍이 등장했다.
- 위처럼 한 CPU가 왔다갔다 하며 일을 처리하는 것이 이를 뜻한다.
- 앞과는 반대로 여러라인을 CPU가 옮겨다닐 수 있다.

자바스크립트에서의 동시성
비동기(asynchronous) vs 동기(synchronous)

- 앞선 코드는 "동기"코드이다.
- 동기코드는 시간이 오래 걸리는 작업이 있을 때 다른 것을 할 수가 없다.
- 그래서 "비동기"라는 것을 사용한다.

자바스크립트에서의 동시성
가장 쉬운 비동기

- 간단하게 비동기를 만들 수 있는 방법은 setTimeout을 사용하는 것이다.
- 이렇게 하면 동시에 일을 처리할 수 있겠죠?
- 동시 ≠ 동기

자바스크립트에서의 동시성
비동기의 활용

- 시간이 걸리는 작업
- 작업을 하면서 리소스를 많이 잡지 않는 작업
- 외부로 요청을 보내는 것:
  - 서버로 요청 보내기 (fetch)
  - DB에서 데이터 가져오기
  - 파일 읽기

자바스크립트에서의 동시성

엥? 그런데 값은 어떻게 받아요?

- return을 사용?
  - 이상한 값이 나온다.
- 그러면 setTimeout안에 setTimeout을 계속 쓰면 될 것 같다.
- ??????????
- 결론은 setTimeout을 하면 값을 받을 수 없다.
- 콜백이라는 개념이 등장

자바스크립트에서의 동시성
콜백

- 1급 함수 특징을 사용해서!

자바스크립트에서의 동시성

너무 복잡해진다...

- 대신에 **"상태"**를 **돌려주게끔** 만들 것이다.
- 앞에서 본 setTimeout은 단순히 **상태**를 만드는 뜻을 한다고 하면, 지금 볼 것은 그 **상태**를 **"값"**으로 만들 것이다.
- 여기서 Promise라는 객체를 쓸 것이다.

자바스크립트에서의 동시성

Promise

- Promise. 한글로 약속.
- 상태를 나타낼 수 있는 **값**이다.
- 세가지 상태가 존재한다.
  - pending: 기다리는 중
  - fulfilled: 성공!
  - rejected: 실패
- 약속이지만, 꼭 성공한다는 보장이 없다고 보면 된다.

자바스크립트에서의 동시성

Promise에서 값을 받아오기

- 콜백이 아닌, then이라는 것을 사용한다.
- then안에 함수를 써주는 방식으로 값을 가져올 수 있다.

자바스크립트에서의 동시성
시간이 남으면

- async, await

자바스크립트에서의 동시성

async, await란?

- then 대신에 사용하는 것이다.
- then으로 하면 단점이 있다. 모아서 값을 전달할 수 없다.
- async와 await를 쓰면 이를 해결할 수 있다.

자바스크립트에서의 동시성

async에 대한 오해

- async가 흔히 비동기 함수를 만드는 키워드라고 생각하기 쉬운데, 아니다!!
- async는 await를 쓰기 위한 것이다.