Armen Melkumyan • 1st
Technical / Solutions Architect
1w •

Interesting Coding Puzzle from a JavaScript/React Technical Interview

Ever wondered how those tricky questions around asynchronous behavior pop up in real interviews? Check out this short snippet that tests your knowledge of JavaScript's event loop—particularly the interplay between synchronous code, microtasks (Promises), and macrotasks (setTimeout):

```
console.log("A");

setTimeout(() => {
 console.log("B");
}, 0);

Promise.resolve().then(() => {
 console.log("C");
 Promise.resolve().then(() => {
 console.log("D");
 });
});

setTimeout(() => {
 console.log("E");
}, 0);

console.log("F");
```

What's the Output?

The correct order is:

A

F

C

D

B

E

Why Does This Happen?

Synchronous Code First

JavaScript runs all synchronous statements in order. So, "A" is logged first, and then immediately after the code block with Promises and timeouts is set up, "F" appears.

Microtasks (Promises) Next

After the synchronous tasks finish, any queued Promise callbacks (microtasks) are processed before the engine moves on to the next macrotask. This is why "C" is logged, followed by "D" from the nested Promise.

Macrotasks (setTimeout) Last

Finally, the setTimeout callbacks are considered macrotasks, so "B" and then "E" are logged in the order they were scheduled—both after all the microtasks have finished.

#React #JavaScript #TechInterview